Inform. Systems Vol. 12, No. 2, pp. 171–175, 1987
Printed in Great Britain

0306-4379/87 $3.00 + 0.00
Pergamon Journals Ltd

VORONOI TREES AND CLUSTERING PROBLEMS

F. DEHNE and H. NOLTEMEIER
Informatik I, University of Würzburg, Am Hubland, D-8700 Würzburg, West Germany

(*Received* 23 *March* 1986; *in revised form* 13 *November* 1986)

Abstract—This paper presents a new data structure called Voronoi trees to support the solution of proximity problems in general quasi-metric spaces with efficiently computable distance functions. We analyze some structural properties and report experimental results showing that Voronoi trees are a proper and very efficient tool for the representation of proximity properties and generation of suitable clusterings.

Key words: Cluster analysis, proximity problems, Voronoi diagram, data structures.

1. INTRODUCTION

Cluster analysis is a familiar technique not only in the field of pattern recognition, but is very important in other domains too, especially in information retrieval. It may be characterized by the use of resemblance or dissemblance measures between objects to be identified. The objective of a cluster analysis is to uncover natural groupings, or types, of objects [1,2].

In this paper we are concerned with the problem of how to support proximity and clustering problems in quasi-metric spaces.

A set E is called *quasi-metric space* if there is a distance function $d:E^2 \to R_+$ with the following properties (for any $e, e', e'' \in E$):

(1) $d(e, e) = 0$;
(2) $d(e, e') = d(e', e)$ (symmetry);
(3) $d(e, e'') \leqslant d(e, e') + d(e', e'')$ (triangle inequality).

However, it may happen in quasi-metric spaces that there are two different elements $e, e' \in E$ with $d(e, e') = 0$. Most of the literature of nearest neighbor problems is dealing with finite dimensional real spaces with some L_p norm—very few are considering more general spaces.

One of these few more general approaches is the recent work of Kalantari and McDonald [3]. The data structure they propose is a straightforward generalization of binary search trees (which can support nearest neighbor search in E^1) and is applicable to any normed space if the norm is computable effectively. The next section will briefly summarize their approach but will also point out some disadvantages. In section 3 we propose a slightly different structure (which we call "Voronoi tree") which, however, will show significant structural advantages. Some experimental results will be stated in Section 4. Due to their structural advantages it turned out that Voronoi trees are a proper and very efficient tool for the representation of proximity properties and solution of clustering problems.

2. BISECTOR TREES

Let E be an arbitrary space, $d:E^2 \to R_+$ a mapping which induces a metric on E, and $S = \{e_1, \ldots, e_n\}$ a finite point set in E [3]. Represent S using a binary tree (called "bisector tree" or "bs-tree" for the remainder of this paper) as follows:

(a) each node of the actual tree T_i (representing the actual set $S_i := \{e_1, \ldots, e_i\}$, $0 \leqslant i \leqslant n$) contains at least one (p_L) and at most two elements (p_L and p_R) of S_i;

(b) to insert a new element e_{i+1} into T_i (yielding T_{i+1}) start at the root node of T_i as follows:

—if the actual node v contains only one element then insert e_{i+1} into v;
—if (otherwise) v contains two elements p_L and p_R then recursively insert e_{i+1} into the left subtree (which has root p_L) if $d(e_{i+1}, p_L) < d(e_{i+1}, p_R)$, or into the right subtree (which has root p_R) if $d(e_{i+1}, p_R) < d(e_{i+1}, p_L)$ and arbitrarily if the distances are equal.

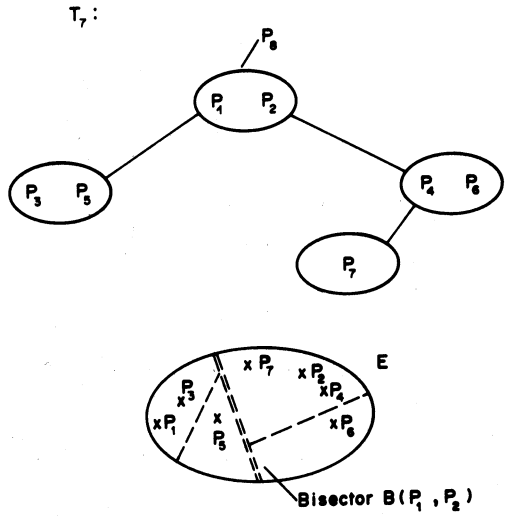

Fig. 1

To support nearest neighbor queries each node v of the bs-tree T storing two elements p_L, p_R with left subtree T_L and right subtree T_R additionaly stores the following two values:

LRADIUS(v): $= \max\{d(p, p_L)/p$ is stored in $T_L\}$ (left radius of v);
RRADIUS(v): $= \max\{d(p, p_R)/p$ is stored in $T_R\}$ (right radius of v).

Thus, given a query point $P \in E$ and a bs-tree representing a set $S \in E$ in order to search for the nearest neighbor of P in S, the search process can prune the left [right] subtree T_L [T_R] of an actual node v if

(*) $d(P, P_L) - \text{LRADIUS}(v) \geq \text{DACTUAL}$
[$d(P, P_R) - \text{RRADIUS}(v) \geq \text{DACTUAL}$]

holds with DACTUAL denoting the distance of P to its actual nearest neighbor in S (for more details see [3]).

However, we have to state the following remarks:

(i) a son may have larger radia than his father ("eccentric son")

(ii) there are cases in which a bs-tree has to be searched exhaustively even if we consider only balanced bs-trees (having logarithmic height).

(It is easy to construct examples for such cases even in E^2 and is left to the reader.)

3. THE VORONOI TREE (VT)

To overcome some disadvantages of bs-trees let us introduce "Voronoi trees". Let E be any quasi-metric space (with distance function d) as described in Section 1, and S be a finite subset of E.

A Voronoi tree (VT) which represents S is a ternary tree with each node prepresenting at least two and at most three points of S—the root node of course additionally is allowed to represent only one element too. To insert a new point into a VT we proceed in essentially the same way as described in Section 2 but with the following difference:

(**) If a new leaf v has to be created in order to store a new point P and Q is P's nearest neighbor of the (three) points stored in the father node of v, then Q (redundantly) has to be stored in v, too (see Fig. 2).

With this we can prove the following

Theorem

(1) Voronoi trees do not have eccentric sons.

(2) A Voronoi tree is *transitive* in the following sense: $d[P,$ father$(P)] = \min\{d(P, P')/P' \in$ ancestors$(P)\}$ [with ancestors(P) denoting the set of all elements stored in those nodes which are on the path from the root to P].

(3) If we search for nearest neighbors of points which are stored in the root node, then this search is possible in time $O(h)$ with h denoting the height of the tree.

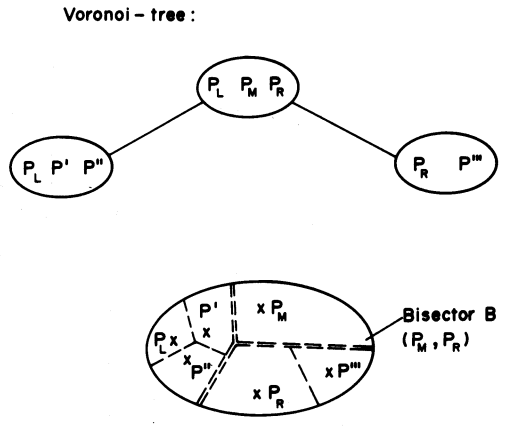

Fig. 2

Remark: The term "Voronoi tree" is chosen due to the fact that all triples of bisectors exactly determine the vertices of all order k Voronoi diagrams (cf. [4,5]).

4. EXPERIMENTAL RESULTS

In this section we will present experimental results to demonstrate that VT-trees can be efficiently used to represent proximity properties in quasi-metric spaces (in general no linear spaces) and especially that VT-trees are an efficient tool for the solution of clustering problems.

Our test data were taken from two very different areas:

(1) randomly generated points in the unit cube of the d-dimensional real space R^d where the metric can be chosen from arbitrary L_p norm (e.g. $d = 2,3,4,5$ and $p = 1,2,\infty$);

(2) a set of 15,000 elements and three different quasi-metrices which were induced by a given Gozinto graph (see [6,7]) alternatively gave the basis for our field experiments.

In each case the Voronoi tree was constructed (with a wide range of different parameters and number of points in the first environment) and the computation time, storage requirement, height of the VT-tree, scattering of the heights of the leafs, etc. was analyzed and compared with experimental results from bs-trees and ternary bs-trees (bs-trees with at most 3 points stored in each node but without condition (**), $k = 3$ in [3]).

Additionally, the VT could be modified by choosing some rule how to place a copy of a father P' in his son node with respect to condition (**), i.e. fixed place, cyclic placement during the construction, randomly chosen, or "at the same place as in the father node" (hereditary property).

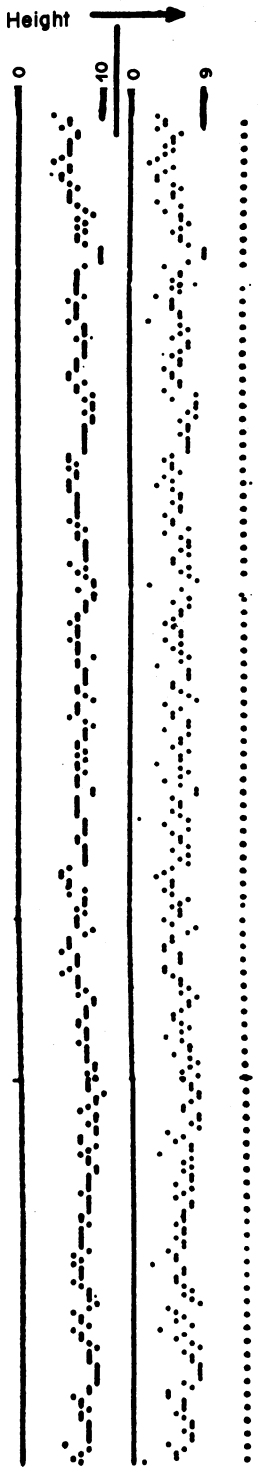

Fig. 3. Voronoi-tree (representing 1000 points from R^2, L_1-metric): Heights of leafs (upper part), heights of first occurrence of the father corresponding to the leaf (lower part).

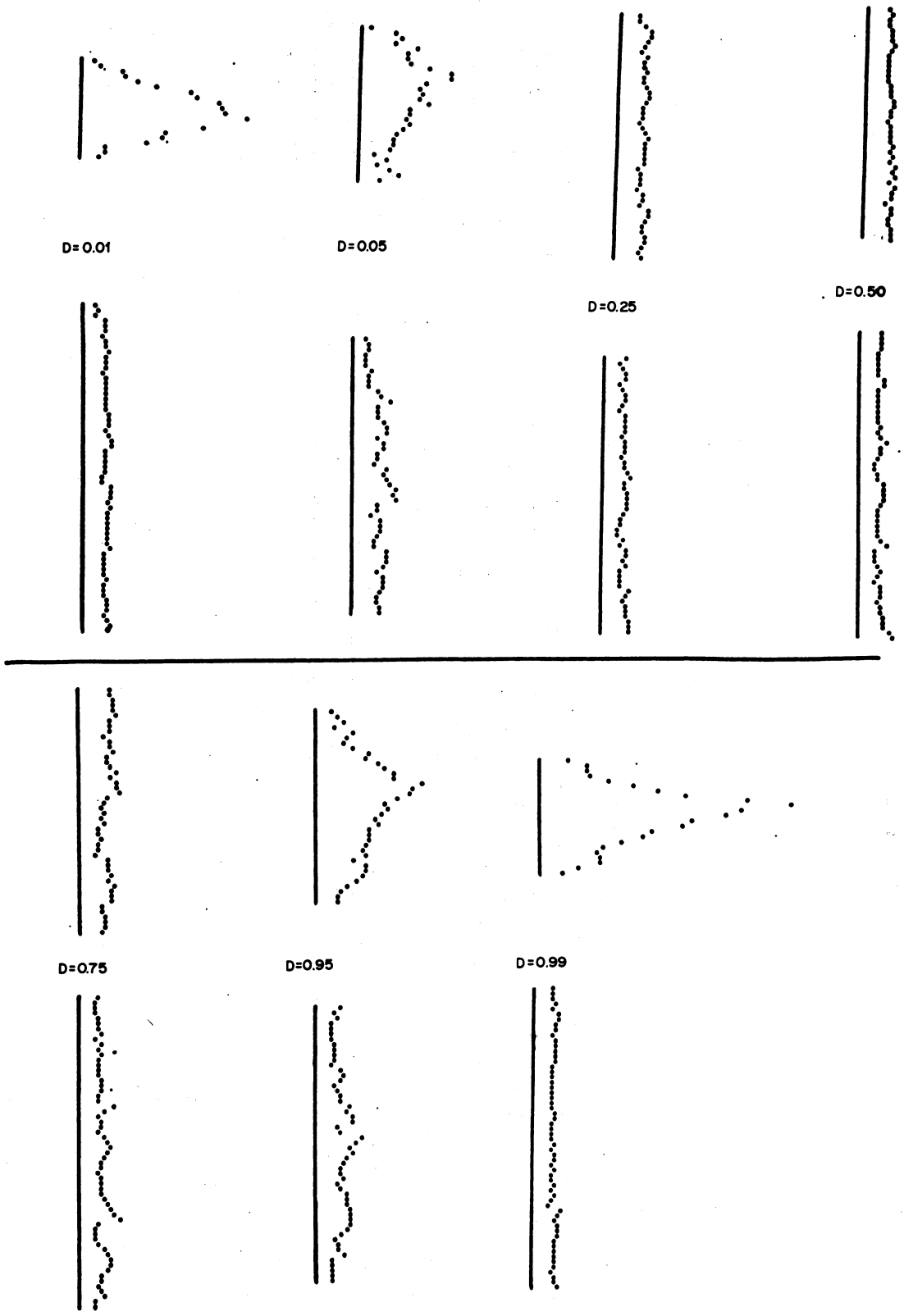

Fig. 4. Comparison of the height of bs-trees (upper-diagram) and Voronoi-trees (lower diagram). D: density of edges in the Gozinto-graph.

Table 1. Gozinto-graph (see [6,7]) containing 14,457 elements, 2259 groups and 54,735 edges (density $D = 0.00168$)

	bs-tree	Voronoi tree[a]
Construction time (CPU, s)	513	35
Number of leafs	1512	4273
Minimum height of leafs	5	5
Maximum height of leafs	1211	16
Average height of leafs	427.27	10.24

[a]With hereditary property.

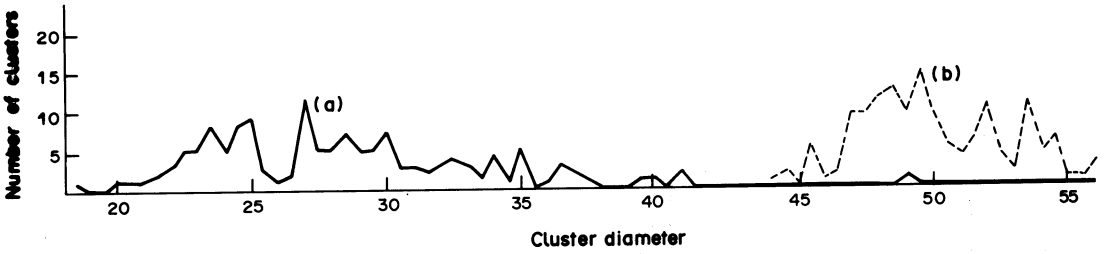

Fig. 5. Distribution of the diameters of clusters induced by Voronoi-trees (a) and randomly generated (b). Sample set: 5000 points in E^5, cluster size between 30 and 50.

We furthermore studied how the order in which the points are inserted into the VT-tree influences the final result, especially some (dynamic) self-organizing principles.

The figures illustrate a small but typical sample from our test series.

Similar to this series (with relatively small sample set) our test series in the second environment showed that the time required to construct a Voronoi tree is much smaller than the time needed to build a bs-tree or ternary bs-tree. This result is due to the fact that Voronoi trees showed to be balanced much better than bs-trees which was expectable from the theorem. Table 1 states some experiences with our large scale real world data (environment 2) and exemplifies our general experimental results.

Applied to our test data the use of Voronoi trees did not only reduce computation time dramatically, but did also induce feasible clusterings of large point sets in quasi-metric spaces. It showed that subtrees of a Voronoi tree represent subsets of points with certain proximity properties (with respect to the chosen quasi-metric of course). How good these proximity properties are represented by Voronoi trees is exemplified in Fig. 5.

REFERENCES

[1] E. Diday and J. C. Simon. Clustering analysis. In K. S. Fu (Ed.), *Digital Pattern Recognition*. Springer, Berlin, (1980).

[2] J. Dubes. Clustering methodologies in exploratory data analysis. In M. C. Yovits (Ed.) *Advances in Computers*, Vol. 19. Academic Press, New York (1980).

[3] I. Kalantari and G. McDonald. A data structure and an algorithm for the nearest point problem. *IEEE Trans. Software Engng* SE-9(5), (1983).

[4] F. Dehne. An $O(n^4)$ algorithm to construct all Voronoi diagrams for K nearest neighbor searching in the euclidean plane. *Proc. 10th Int. Collo. on Automata, Languages and Programming* (*ICALP* '83), Barcelona. Lecture Notes in Computer Science, No. 154. Springer, Heidelberg (July, 1983).

[5] M. I. Shamos and K. Hoey. Closest point problems. *Proc. 16th Ann. IEEE Symp. on Foundations of Computer Science* (1975).

[6] F. Dehne and H. Noltemeier. Clustering methods for geometric objects and applications to design problems. *The Visual Computer*, Vol. 2. Springer, Heidelberg (1986).

[7] H. Noltemeier. Distances in hypergraphs, Report. Informatik I, Würzburg (1985).

[8] F. Dehne and H. Noltemeier. Clustering geometric objects and applications to layout problems. *Proc. "Computer Graphics 1985"*, Tokyo. Springer, Tokyo (1985).

[9] F. Dehne and H. Noltemeier. A computational geometry approach to clustering problems. *Proc. 1st ACM SIGGRAPH Symp. on Computational Geometry*, Baltimore. (June, 1985).

[10] F. Dehne. Optical Clustering, Report. Informatik I, Würzburg (1985).