Revised version, February 1991, appeared in *Information Processing Letters* **38** (1991), 123–127

COMPUTING THE MINIMUM HAUSDORFF DISTANCE BETWEEN TWO POINT SETS ON A LINE UNDER TRANSLATION

Günter Rote
Technische Universität Graz, Institut für Mathematik, Steirergasse 30, A-8010 Graz, Austria.
E-mail: rote@ftug.dnet.tu-graz.ac.at

Given two sets of points on a line, we want to translate one of them so that their Hausdorff distance (the maximum of the distances from a point in any of the sets to the nearest point in the other set) is as small as possible. We present an optimal $O(n \log n)$ algorithm for this problem.

Keywords: computational geometry, Hausdorff distance, pattern recognition, pattern matching.

1 Introduction

The *Hausdorff distance* between two given sets of numbers $A = \{a_1, a_2, \ldots, a_n\}$ and $B = \{b_1, b_2, \ldots, b_m\}$ is defined as

$$d(A,B) = \max\Big\{\max_{a\in A}\min_{b\in B}|b-a|,\ \max_{b\in B}\min_{a\in A}|a-b|\Big\}.$$

In other words, the Hausdorff distance between A and B is the smallest value d such that every point of A has a point of B within distance d and every point of B has a point of A within distance d. The Hausdorff distance can also be defined for point sets in two or more dimensions, where $|a-b|$ must be replaced by the Euclidean distance between a and b (or any other appropriate distance function).

In applications like computer vision or pattern recognition, the set A might represent an "image" taken by a camera and B might represent a "template" against which A should be matched. Usually the relative position of the image A is not fixed with respect to the coordinate system in which the template is given: we may translate A so that it becomes aligned with B and matches B as well as possible.

In this paper, we solve the one-dimensional version of the problem for two point sets on a line (see figure 1, where the point sets are represented as the teeth of two combs). Formally, we compute

$$\min_{t\in I\!R} d(A+t,B) = \min_{t\in I\!R}\ \max\Big\{\max_{a\in A}\min_{b\in B}|b-(a+t)|,\ \max_{b\in B}\min_{a\in A}|(a+t)-b|\Big\}.$$

Here the parameter t denotes the amount by which A is shifted. The algorithm takes $O((m+n)\log(m+n))$ time and is described in the next section. Previously, the problem could only be solved in $O(mn\log(m+n))$ time, see Huttenlocher and Kedem [5]. The concluding section discusses implementation issues, a lower bound for the time complexity, the limitations of our method, and some related problems.

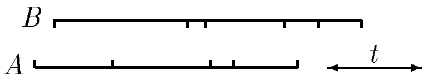

Figure 1: Two sets of points.

2 The algorithm

In a first step we (conceptually) transform the problem to a variation dealing with the "non-symmetric" or "one-way" Hausdorff distance:

$$\tilde{d}(\mathcal{A}, \mathcal{B}) = \max_{a \in \mathcal{A}} \min_{b \in \mathcal{B}} |b - a|.$$

This will make the algorithm more convenient to explain, because we can now treat the sets $\mathcal{A}$ and $\mathcal{B}$ differently without confusion. To transform the original problem, we take a reflected copy $\tilde{A}$ of A and add it to the right of B, sufficiently far apart (see figure 2). (Any distance three times larger than the diameter of A plus the diameter of B would certainly be sufficient.) Similarly, we take a reflected copy $\tilde{B}$ of B and add it to A on the right side, the same distance apart. The new sets are called $\mathcal{B}$ and $\mathcal{A}$, respectively; their cardinality is $m + n$. It is clear that the non-symmetric Hausdorff distance of the new sets is the same as the Hausdorff distance of the two original sets, and this equality remains true if we translate any of the sets. Figure 2 shows the Hausdorff distance of the new sets and where it occurs.

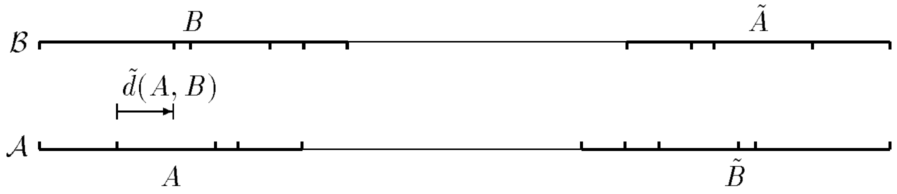

Figure 2: Extending the original sets.

So we now want to minimize the function $F(t) := \tilde{d}(\mathcal{A} + t, \mathcal{B})$. This function can be built up in the following way: For a single point x, $f(x) = \tilde{d}(x, \mathcal{B})$ is a continuous piecewise linear function of the variable x (see figure 3), with edges of slope 1 and -1. Its minima are at the points of $\mathcal{B}$. We have to consider $m + n$ such functions $f(a + t)$, which are basically the same, except that they are shifted in the horizontal direction by different amounts a. The function $F(t)$ is just the upper envelope of these functions:

$$F(t) = \max_{a \in A} f(a + t) \tag{1}$$

Like $f(x)$, the function $F(t)$ is piecewise linear, with edges of slope ± 1. Each edge of $F(t)$ originates from (at least) one of the functions $f(a + t)$ in (1). It is straightforward to construct F in $O((m+n)^2 \log(m+n))$ time by varying t from $-\infty$ to $+\infty$ and maintaining the current order of the $m + n$ values $f(a + t)$. (Since we are only interested in distances from A to B and from $\tilde{B}$ to $\tilde{A}$, not from A to $\tilde{A}$ or from $\tilde{B}$ to B, our original problem can actually be solved in $O(mn \log(m + n))$ time, see Huttenlocher and Kedem [5]).

However, the following considerations show that we do not need so much effort to construct the function F. We assume that the initial position of $\mathcal{A}$ and $\mathcal{B}$ (with parameter

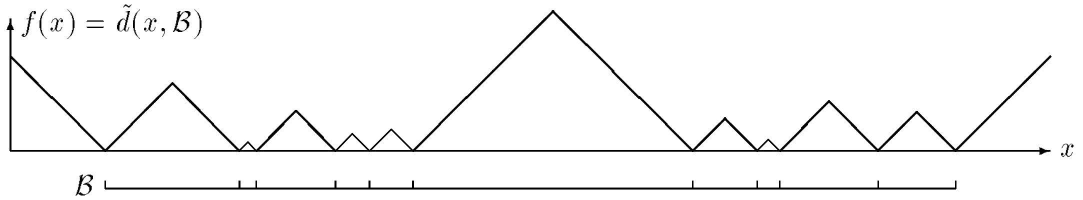

Figure 3: The function $f(x) = \tilde{d}(x, \mathcal{B})$

value $t = 0$) corresponds to the position where $\mathcal{A}$ and $\mathcal{B}$ are aligned at their endpoints, as in figure 2.

If we shift $\mathcal{A}$ by t, the left or right endpoint will contribute at least $|t|$ to the Hausdorff distance, and so we know that $F(t) \geq |t|$. From this lower bound we can conclude:

Lemma *Each of the $m + n$ functions $f(a + t)$, for $a \in \mathcal{A}$, in equation (1) contributes at most two edges to the function $F(t)$.*

Proof. Consider some function $f(a + t)$ that has a local maximum at some point t_0 with $f(a + t_0) > |t_0|$, (see figure 4). From this maximum the function decreases to 0 towards both sides before it can rise again. Since the slope is always ± 1, the only two edges that appear above the lower bound $F(t) \geq |t|$ are the two edges emanating from t_0. In case $f(a + t)$ does not have such a local maximum only one edge can appear above the lower bound $F(t) \geq |t|$. (In the exceptional case that $f(a) = 0$, two edges may form part of the lower bound, and the lemma is also true.)

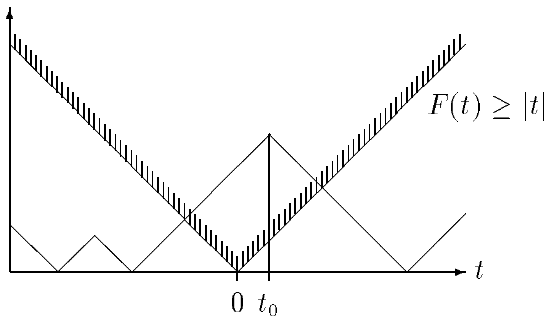

Figure 4: A lower bound for $F(t)$ and one function $f(a + t)$.

The lemma implies that $F(t)$ has at most $2(m+n)$ linear pieces. (A more exact analysis shows that the true (tight) upper bound on the number of edges is $2(m + n) - 8$, for all $m \geq 3$ and $n \geq 3$.)

To find the two pieces of $f(a + t)$ which possibly contribute to $F(t)$ we compute the function at $t = 0$, (i. e., we compute $f(a)$) and we follow the edge in the rising direction to the next peak, which is the local maximum of the lemma, from which the two relevant edges start. The other cases of the lemma are also handled easily. If the function $f(x)$ is given as a sorted list of the points in $\mathcal{B}$ and the functions $f(a+t)$ are processed in order of increasing values $a \in \mathcal{A}$, the $2(m + n)$ linear pieces can be found in $O(m + n)$ time. (The process is essentially a merging of $\mathcal{A}$ and $\mathcal{B}$.)

Now, $F(t)$ is the (pointwise) maximum of these $2(m + n)$ linear pieces of slopes ± 1 (see figure 5). $F(t)$ can easily be computed (and minimized) in $O((m + n) \log(m + n))$ time: We could for example add the functions $f(a + t)$ one at a time and maintain the upper envelope of the functions added so far in a balanced tree. Alternatively, we could sort the "spikes" (where a spike consists of a rising edge, a local maximum, and a falling edge) according to their left endpoints and add them from left to right (cf. Huttenlocher and Kedem [5], algorithm 2). After the sorting, this can be carried out in linear time.

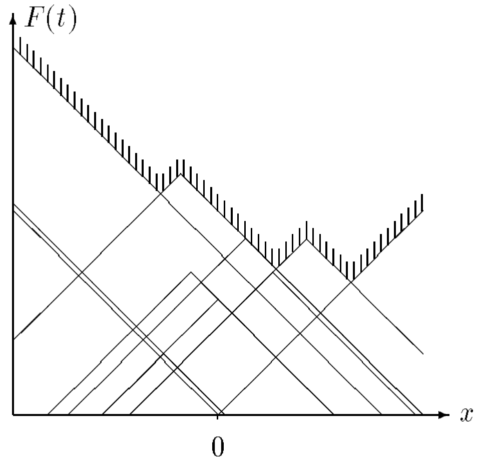

Figure 5: The upper envelope $F(t)$ of the relevant part of the functions $f(a+t)$, for $a \in \mathcal{A}$, for the example of figure 2.

Theorem *The minimal Hausdorff distance between two point sets on a line under translation can be computed in $O(n \log n)$ time and $O(n)$ space, where n is the total number of given points.*

Proof. As was discussed in the paragraph preceding the theorem, the two relevant edges for each of the n functions $f(a+t)$ can be found in linear time if $\mathcal{B}$ and $\mathcal{A}$ are sorted, and their upper envelope can be constructed in $O(n \log n)$ time. Scanning this upper envelope to find its minimum takes again only linear time. It is clear that linear space is sufficient for all steps.

3 Conclusion

Note that apart from two sorting steps, all steps of our algorithm can be carried out in linear time. The two sorting steps that are involved are the sorting of the initial data a_i and b_j and one sorting step for constructing $F(t)$, where certain numbers of the form $b_j - a_i$ have to be sorted. This is of interest because in practice the time complexity of sorting is usually a "fast" $O(n \log n)$ (with a low constant in the O-notation); therefore our algorithm should be very fast, too.

The complexity of $O(n \log n)$ of our algorithm is optimal in the algebraic decision tree model, *even if the initial data a_i and b_j are given in sorted oder.* This can be shown with the technique of Ben-Or [1] (see [7, p. 30]). For a sketch of the proof we refer to figure 5. If all spikes of the functions $f(a+t)$ have the same height, minimizing their upper envelope $F(t)$ corresponds to finding the maximum gap between adjacent spikes. For this so-called MAX GAP problem an $\Omega(n \log n)$ lower bound has been proved by Ramanan [8] and Lee and

Wu [6, appendix]. The left endpoints of the spikes are numbers of the form $b_j - a_i$. It is not too difficult to convince oneself that an arbitrary given instance of the MAX GAP problem can be transformed in linear time into two sorted sets of numbers $A = \{a_1, a_2, \ldots, a_n\}$ and $B = \{b_1, b_2, \ldots, b_n\}$ for our problem so that the spikes of the function F are at the desired positions.

The MAX GAP problem can be solved in linear time if the floor function $\lfloor \cdot \rfloor$ can be used (see [7, pp. 253–254]). However, since the spikes of the functions $f(a + t)$ need not have the same height, our problem is actually more general than the MAX GAP problem. For example, if we are given the spikes of F and want to decide whether there is a point t with $F(t) \leq d$, for some threshold d, we have to determine whether a set of intervals is disjoint. Thus we actually have a max gap problem for *intervals*. This makes it unlikely that the lower bound for our problem can be overcome even if the floor function is used.

Although we have reduced our problem to the problem with the non-symmetric Hausdorff distance, our method cannot be applied if the problem is originally posed for two arbitrary given sets with the non-symmetric Hausdorff distance. The reason is that a crucial requirement for our method is that the diameter of $\mathcal{A}$ is at least as big as the diameter of $\mathcal{B}$, which implies the lower bound $F(t) \geq |t|$. Huttenlocher and Kedem [5] give an example where the function F for the non-symmetric Hausdorff distance can have $\Omega(mn)$ breakpoints, all at the same height. Our method does therefore not provide a way to avoid computing all of these breakpoints and to improve the $O(mn \log(m + n))$ bound for this general problem.

In one of the applications of Huttenlocher and Kedem [5], the sets A and B were actually not points on a line, but points on the unit circle, and the allowed "translation" was accordingly a rotation about the origin. In this case our algorithm also does not work, for the same reason as above.

A related problem occurs in certain scheduling problems for urban transportation systems: When two periodic railway lines share the same track, the safety distance between successive trains should be as large as possible. Finding the appropriate amount by which the schedule of one line should be shifted is equivalent to *maximizing* the Hausdorff distance between two point sets on a circle under rotation. Brucker and Meyer [3] solve this problem in $O(mn \log(m + n))$ time. The problem can of course be generalized to more than two train lines, and it has also some variations, for example, when several bus lines meet at a terminal and the maximum (or average) waiting time for the passengers should be minimized. These problems lead to various different objective functions, some of which are similar to the Hausdorff distance, see Brucker, Burkard, and Hurink [2] or Burkard [4].

References

[1] M. Ben-Or, Lower bounds for algebraic computation trees, in: *Proc. 15th Annual ACM Symp. on Theory of Computing*, pp. 80–86, 1983.

[2] P. Brucker, R. E. Burkard, and J. Hurink, Cyclic schedules for r irregularly occurring events, *J. Computational and Applied Mathematics* **30** (1990), 173–189.

[3] P. Brucker and W. Meyer, Scheduling two irregular polygons, *Discrete Applied Mathematics* **20** (1988), 91–100.

[4] R. E. Burkard, Optimal schedules for periodically recurring events. *Discrete Applied Mathematics* **15** (1986), 167–180.

[5] D. P. Huttenlocher and K. Kedem, Computing the minimum Hausdorff distance for point sets under translation, in: *Proceedings of the Sixth Annual Symposium on Computational Geometry*, Berkeley, California, June 6–8, 1990. Association for Computing Machinery, 1990; pp. 340–349.

[6] D. T. Lee and Y. F. Wu, Geometric complexity of some location problems, *Algorithmica* **1** (1986), 193–212.

[7] F. P. Preparata and M. I. Shamos, *Computational Geometry: An Introduction*, Springer-Verlag, 1985.

[8] P. Ramanan, Obtaining lower bounds using artificial components, *Information Processing Letters* **24** (1987), 243–246.