アルゴリズム 8－4
（1989. 6. 23）

幾つかの最適化基準の下での2次元点集合のクラスタリング／ハッシング算法

浅野哲夫[1]　今井浩[2]　今井桂子[3]
[1]大阪電気通信大学　[2]九州大学工学部
[3]九州工業大学

本論文では、平面上の点集合に対するクラスタリング及びハッシングに関連する幾つかの問題について効率のよいアルゴリズムを提案する。考察する最初の問題は、最小ギャップの相対値が最大になるように与えられた点集合を直線上に射影する問題である。この他、最大ギャップの相対値を最小にするような射影を求める問題についても考察する。これらの問題に対して、それぞれ平面走査法と位相的走査法に基づく効率のよいアルゴリズムを提案するが、いずれも双対平面上での直線の配置構造における壁と区間の概念の基づいているのが特徴である。本論文で提案されるアルゴリズムはすべて線形の記憶量しか必要としない。

Clustering/Hashing Points in the Plane with Maxmin Criteria

Tetsuo Asano*, Hiroshi Imai** and Keiko Imai***
* Osaka Electro-Communication University, Japan
** Kyushu University, Japan,
*** Kyushu Institute of Technology, Japan.

ABSTRACT

This paper presents efficient algorithms for several problems related to clustering and hashing points in the plane. One of the problems is to find a projection to maximize the minimum relative gap among projected points. Another problem is to maximize the maximum relative gap. The former problem can be solved by a plane sweep method and the latter one by a topological sweep, both based on the ideas of wall and interval in the arrangement of lines in the dual space. All the algorithms presented run in linear space.

1 Introduction

This paper considers several optimization problems related to clustering and hashing points in the plane. We are given a set S of n points in the plane. For θ with $0 \leq \theta < \pi$, define $z_i(\theta)$ by

$z_i(\theta) = x_i \cos\theta + y_i \sin\theta,$

and denote the jth smallest value among $z_i(\theta)(i = 1, \ldots, n)$ by $z'_j(\theta)(j = 1, \ldots, n)$ and define the width $w(\theta)$ and the internal maximum and minimum gap $c_{max}(\theta)$ and $c_{min}(\theta)$, respectively, as follows:

$w(\theta) = z'_n(\theta) - z'_1(\theta),$

$c_{max}(\theta) = \max_{i=1,\ldots,n-1}(z'_{i+1}(\theta) - z'_i(\theta)),$

$c_{min}(\theta) = \min_{i=1,\ldots,n-1}(z'_{i+1}(\theta) - z'_i(\theta)).$

The problem of minimizing $w(\theta)$ for $0 \leq \theta < \pi$ is the well-known width (or, linear approximation) problem for points, and can be solved in $O(n \log n)$ time. Considering the two extreme gaps defined above, there arise the following problems:

P1: $\max_{0 \leq \theta < \pi} \frac{c_{min}(\theta)}{w(\theta)}$

P2: $\max_{0 \leq \theta < \pi} c_{min}(\theta)$

P3: $\max_{0 \leq \theta < \pi} c_{max}(\theta)$

P4: $\max_{0 \leq \theta < \pi} \frac{c_{max}(\theta)}{w(\theta)}$

P5: $\min_{0 \leq \theta < \pi}(w(\theta) - c_{max}(\theta))$

The problem P1 was originally motivated by Sprugnoli [Sp77] in connection with perfect hashing functions for a static set of keys. The first efficient algorithm for this problem was given by Comer and O'Donnell [CO82]. Their algorithm runs in $O(n^2 \log n)$ time using $O(n^2)$ space. Algorithmically, a main problem in solving P1 is the problem P2. In this paper we present a linear-space algorithm.

The problem P3 is the linear maxmin approximation (cutting) problem. The problems P4 and P5 are its variants, and may be useful in clustering two sets of points which are approximated well by two parallel lines. All the problems may be regarded as a problem on clustering/hashing points in the plane under some maxmin criteria. The results obtained are as follows:

P1, P2: an $O(n^2 \alpha(n) \log^3 n)$ time and linear space algorithm, where $\alpha(n)$ is the functional inverse of Ackermann function and grows very slowly (almost constant). This is less efficient in time but much more efficient in space compared with the existing algorithm.

P3, P4, P5: an $O(n^2)$ time and linear space algorithm.

2 Maximizing the Minimum Relative Gap

We are given a set S of n points $p_i = (x_i, y_i)$ in the plane. For θ with $0 \leq \theta < \pi$ define $z_i(\theta)$ by

$z_i(\theta) = x_i \cos\theta + y_i \sin\theta$

and denote the jth smallest value among $z_i(\theta)(i = 1, \ldots, n)$ by $z'_j(\theta)(j = 1, \ldots, n)$. The maximum distance between two projected points is called the width of the point set and denoted by $w(\theta)$ while the minimum distance is called the minimum gap and denoted by $c_{min}(\theta)$. Formally, they are defined as follows:

$w(\theta) = z'_n(\theta) - z'_1(\theta),$

$c_{min}(\theta) = \min_{i=1,\ldots,n-1}(z'_{i+1}(\theta) - z'_i(\theta))$.

An example is shown in Fig. 1.

In this paper we represent a projection line by its slope instead of its angle θ. Let $y = -(1/a)x$ be a projective line with $a \neq 0$. By $L(x_i, y_i, a)$ we denote the line passing through the point (x_i, y_i) and perpendicular to the line $y = -(1/a)x$, that is, the line $y = a(x - x_i)$. The projection of a point (x_i, y_i) onto the line is the intersection between the two lines $y = -(1/a)x$ and $L(x_i, y_i, a)$. It is easily seen that $L(x_i, y_i, a)$ intersects the y-axis at $y = y_i - ax_i$ and the distance between two projective points for (x_i, y_i) and (x_j, y_j) is proportional to that between those corresponding intersections with the y-axis [Ed87].

Our method is based on a dual transformation from a set of points $\{(x_i, y_i) \mid i = 1, \ldots, n\}$ to a set of lines $\{b = y_i - ax_i \mid i = 1, \ldots, n\}$ in the ab-plane. It is well known that for an arbitrary value of a the vertical distance between two lines $b = y_i - ax_i$ and $b = y_j - ax_j$ is proportional to the distance between corresponding intersections with the y-axis.

Let A be the arrangement of above-mentioned lines. Consider a vertical line $a = a^*$. The line intersects every line $b = y_i - ax_i$ at $b = b_i(a^*) = y_i - a^*x_i$. Let $(b_1(a^*), b_2(a^*), \ldots, b_n(a^*))$ be a sorted sequence of those ordinates in the increasing order. Then, for $\theta = \tan^{-1}(-1/a^*)$ the width $w(\theta)$ and the minimum gap $c_{min}(\theta)$ are given by $(b_n(a^*) - b_1(a^*))\sin\theta$ and $(\min_{j=1,\ldots,n-1}(b_{j+1}(a^*) - b_j(a^*))\sin\theta)$, where $\theta > 0$ is assumed (see Fig. 2). Thus, we have

$$\frac{c_{min}(\theta)}{w(\theta)} = \frac{\min_{j=1,\ldots,n-1}(b_{j+1}(a^*) - b_j(a^*))}{b_n(a^*) - b_1(a^*)}$$

Therefore, it is convenient to define $w(a)$ and $c_{min}(a)$ by

$w(a) = b_n(a) - b_1(a)$, and

$c_{min}(a) = \min_{j=1,\ldots,n-1}(b_{j+1}(a) - b_j(a))$.

Then, we have

$w(\theta) = w(a)\sin\theta$, and

$c_{min}(\theta) = c_{min}(a)\sin\theta$, where $\theta = \tan^{-1}(-1/a)$.

In the arrangement A of n lines as above the kth wall is a sequence of edges which can be joined to the top wall (upper envelope of the arrangement) by a vertical line segment intersecting $k-1$ lines in between them. The region bounded by kth and $(k+1)$st walls in the arrangement is called the kth interval. For each interval I_i we define its silhouette $SIL_i(a)$ to be a function of a which gives the vertical width of the interval for each value of a. As is easily seen, we have

$c_{min}(a) = \min_{j=1,\ldots,n-1} SIL_j(a)$, and

$c_{max}(a) = \max_{j=1,\ldots,n-1} SIL_j(a)$.

Thus, we can find an optimal value of a to maximize the minimum gap $c_{min}(a)$ by sweeping the arrangement while keeping the lowest silhoette. Since the maximum value of $c_{min}(a)$ is achieved at an intersection between silhouettes, i.e., when two (not necessarily adjacent) intervals become of equal length, a topological sweep [EG86] cannot be applied. On the other hand, the maximum value of the maximum gap $c_{max}(a)$ can be found by a topological sweep since it is achieved at a vertex of the arrangement.

Fig. 3 illustrates the notion of walls and silhouette.

We build a binary tree I-TREE as follows. Leaf nodes of I-TREE store the silhouette of intervals defined above. Then their parent nodes keep the lower envelope of the silhouette stored in their sons. Thus, the root keeps the lower envelope of silhouettes of all the intervals.

Our basic strategy is a plane sweep. Thus, we proceed a vertical sweep line from left to right. We do

not stop at every intersection between two different silhouettes since there may be $O(n^3)$ intersections. Suppose that we have only four intervals I_1, I_2, I_3, and I_4. Then, in our algorithm we first compute the lower envelope of SIL_1 and SIL_2 and that of SIL_3 and SIL_4. The lower envelope of the two resulting silhouettes is computed. Thus, every intersection between SIL_1 and SIL_2 is visited while some intersections between SIL_1 and SIL_3 may not be visited if they do not appear in the final lower envelope.

The sweep line stops at every intersection between silhouette for two brother nodes in I-TREE (referred to as *balancing points*) and every vertex at which the width of some interval becomes zero, that is, upper and lower edges of the interval intersect (referred to as *zero points*). Note that zero points corresponds to vertices in the arrangement.

It is shown below that the number of zero points is $O(n^2)$ and that of balancing points is bounded above by $O(n^2\alpha(n)\log n)$. If we could implement operations needed for each zero or balancing point in $O(\log^2 n)$ time, we would have an algorithm which runs in $O(n^2\alpha(n)\log^3 n)$ time.

[Lemma 1] There are at most $O(n^2\alpha(n)\log n)$ balancing points in total.
Proof: Let $v_1, v_2, \ldots, v_{n-1}$ be leaf nodes of I-TREE corresponding to intervals $I_1, I_2, \ldots, I_{n-1}$. Let u be an arbitrary internal node of I-TREE such that the set of leaf nodes in its descendents is $\{v_l, v_{l+1}, \ldots, v_r\}$. We denote the number of vertices in the silhouette SIL_v by $c(v)$. The lemma follows from the following three observations.
Observation 1 I-TREE has $O(\log n)$ levels.
Observation 2 $c(u) \leq \alpha(n')(c(v_l) + c(v_{l+1}) + \cdots + c(v_r))$, where $n' = c(v_l) + c(v_{l+1}) + \cdots + c(v_r)$.
Observation 3 $c(v_1) + c(v_2) + \cdots + c(v_{n-1}) = O(n^2)$.

Observation 3 follows from the fact that each intersection in the arrangement of n lines appears as vertices of silhouettes of three consecutive intervals.

Observation 2: It is not so hard to see that the silhoette SIL_u for the internal node u is equal to the lower envelope of $SIL_{v_l}, SIL_{v_{l+1}}, \ldots, SIL_{v_r}$. Since the lower envelope for an arbitrary set of m line segments in the plane consists of $O(m\alpha(m))$ vertices, we have the observation. □

The following is a detailed description of the algorithm. First of all we describe data structures: two heaps, B-HEAP and Z-HEAP and a balanced binary tree I-TREE.

B-HEAP is a heap to store the value of a together with two intervals at which the lengths of those intervals become equal to each other (called *balancing point*).

Z-HEAP is a heap to store the value of a together with an interval at which the length of the interval becomes zero (called *zero point*).

I-TREE is a balanced binary tree which has all the current intervals at its leaf nodes in the vertical order. The parent of two intervals is the interval having a shorter length. Thus, the root is the interval having the shortest length.

LOW is the name of the current line on the lower envelope.

UP is the name of the current line on the upper envelope.

MaxRatio stores current maximum ratio of the resolution divided by the span.

(input) $S = \{(x_i, y_i) \mid i = 1, \ldots, n\}$.

MaxRatio is set to positive infinity.

Map each point (x_i, y_i) to a line $s_i : b = y_i - ax_i$.

Sort the set of lines in the increasing order of $(-x_i, y_i)$.

Let $(s_1, s_2, \ldots, s_n)$ be the resulting sorted sequence.

Define $n-1$ intervals $I_1 = (s_1, s_2), I_2 = (s_2, s_3), \ldots, I_{n-1} = (s_{n-1}, s_n)$ with lengths $d(I_i) = y_i - y_{i+1} - a(x_i - x_{i+1}), i = 1, \ldots, n-1$.

For each interval I_i

Compute the value of a such that $d(I_i) = 0$.

Put a tuple $(a, (s_i, s_{i+1}))$ into Z-HEAP.

(if $d(I_i) \neq 0$ for any value of a then skip the operation).

Construct I-TREE:

Let $I_1, I_2, \ldots, I_{n-1}$ be leaves of I-TREE.

Build I-TREE level by level. The basic operation is the comparison of the lengths of two intervals I_i and I_j where a is negative infinity. The comparison is done as follows.

/* Note that the difference of the lengths is given by $d(I_i) - d(I_j) = y_i - y_{i+1} - y_j + y_{j+1} - a(x_i - x_{i+1} - x_j + x_{j+1})$*/.

if $x_i - x_{i+1} - x_j + x_{j+1} > 0$ then $d(I_i) > d(I_j)$

else if $x_i - x_{i+1} - x_j + x_{j+1} = 0$ and $y_i - y_{i+1} - y_j + y_{j+1} > 0$ then $d(I_i) > d(I_j)$

else $d(I_i) \leq d(I_j)$.

Also, compute the value of a such that $d(I_i) = d(I_j)$ holds.

if such a value exists then

if $d(I_i) < d(I_j)$ then put (a, I_i, I_j) into B-HEAP

else put (a, I_j, I_i) into B-HEAP.

Let $UP = s_1$ and $LOW = s_n$.

while(B-HEAP or Z-HEAP is not empty)

Compare the minimum elements of B-HEAP and Z-HEAP.

if B-HEAP < Z-HEAP then {

Extract the minimum element (a, I_i, I_j) from B-HEAP.

Replace I_i with I_j in the nodes except the leaves.

Then, if the root is replaced, compute the ratio

ratio $= d(I_i)/d(UP, LOW)$

and if it is less than *MaxRatio* then update *MaxRatio*, where $d(UP, LOW)$ is the distance between UP and LOW.

Delete all the elements in B-HEAP which contain I_i.

For each node of I-TREE that contains I_j,

Find its brother node I_k and compute the balancing point between I_j and I_k to put it into B-HEAP.

}

if Z-HEAP < B-HEAP then {

Extract the minimum element (a, I_i) from Z-HEAP.

Let I_i be (s_1, s_2).

if s_1 lies in the upper envelope then let UP be s_1.
if s_2 lies in the lower envelope then let LOW be s_2.
Find the upper interval $I_u = (s_0, s_1)$ and the lower interval $I_l = (s_2, s_3)$.
Then, the intervals $(s_0, s_1), (s_1, s_2)$, and (s_2, s_3) should be replaced in I-TREE with (s_0, s_2), (s_2, s_1) and (s_1, s_3), respectively, while computing the corresponding balancing points.
Delete all the elements containing those old intervals from B-HEAP and Z-HEAP.
For the new intervals $(s_0, s_2), (s_2, s_1)$, and (s_1, s_3), compute its zero point to be put into Z-HEAP.

}
}

[Theorem 1] It is possible to find an optimal projection to maximize the minimum relative gap $c_{min}(\theta)/w(\theta)$ in $O(n^2\alpha(n)\log^3 n)$ time using linear space.
(Proof) The correctness of the algorithm might be obvious from the above description. The computational complexity follows from the fact that there are $O(n^2)$ zero points and $O(n^2\alpha(n)\log n)$ balancing points and operations for each such point can be done in $O(\log^2 n)$ time. Since there is at most one balancing point and at most one zero point for each node of I-TREE, the space needed is linear in the number of nodes of I-TREE, which is $O(n)$.□

3 Maximizing the Maximum Relative Gap

The problem P3 can be solved in $O(n^2)$ time and linear space using the topological sweep. The problems P4 and P5 also can be solved by the topological sweep while a naive application leads to $O(n^2 \log n)$ time since we have to know the corresponding value of $w(\theta)$. We can modify the topological sweep by using the ideas of wall and interval defined above.

First of all, we compute the upper and lower envelopes of the arrangement of lines in the dual plane. Then, we partition the region bounded by those envelopes into vertical slabs by vertical lines through vertices on the envelopes. Note that the number of those slabs is $O(n)$.

Next, we perform a topological sweep. Here for each interval of the arrangement we prepare an pointer to the sequence of slabs. Thus, the elementary step of the sweep at an intersection v of the arrangement consists of
(1) computing the vertical distances to the lines immediately above and below the point v (let intervals above and below v be $(i-1)$-st and $(i+1)$-st intervals),
(2) computing the relative gaps (width of the interval divided by the width of the corresponding vertical slab) in each of the intervals since the slab pointed by the pointer of that interval until the slab including the point v, and
(3) updating the pointers.

In other words we merge slabs with each interval. The merging process for some interval may require quadratic time while we need only $O(n)$ extra time for the process in each interval other than the time required by the topological sweep. Therefore, the time required is $O(n^2)$ in total.

REFERENCES

[CO82] D. Comer and M. J. O'Donnell: *Geometric Problems with Applicatios to Hashing*, SIAM J. Comput., vol.11, No.2,pp.217-226, 1982.

[Ed87] H. Edelsbrunner, *Algorithms in Combinatorial Geometry*, EATCS Monographs in Theoretical Computer Science, vol. 10, Springer-Verlag, Berlin, Heidelberg, 1987.

[EG86] H. Edelsbrunner and L.J. Guibas, *Topologically Sweeping an Arrangement*, Proc. 18th Ann. ACM Symp. Theory of Computing, pp. 389-403, 1986.

[Mat88] J. Matousek: *Line Arrangements and Range Search,* Inform. Proc. Lett., 27, pp.275-280, 1988.

[Go83] G.T. Toussaint, *Solving Geometric Problems with the "Rotating Calipers"*, Proceedings of IEEE MELECON '83, Athens, Greece, 1983.

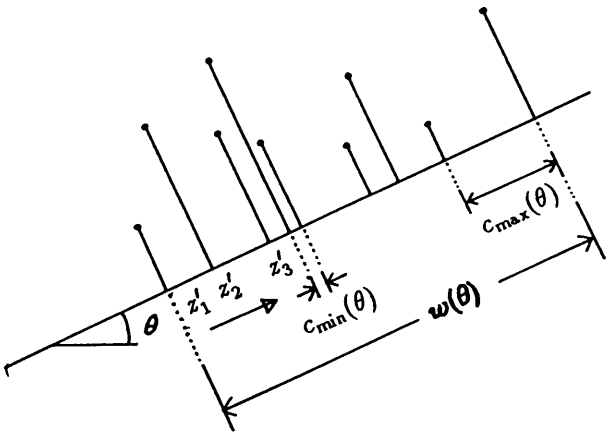
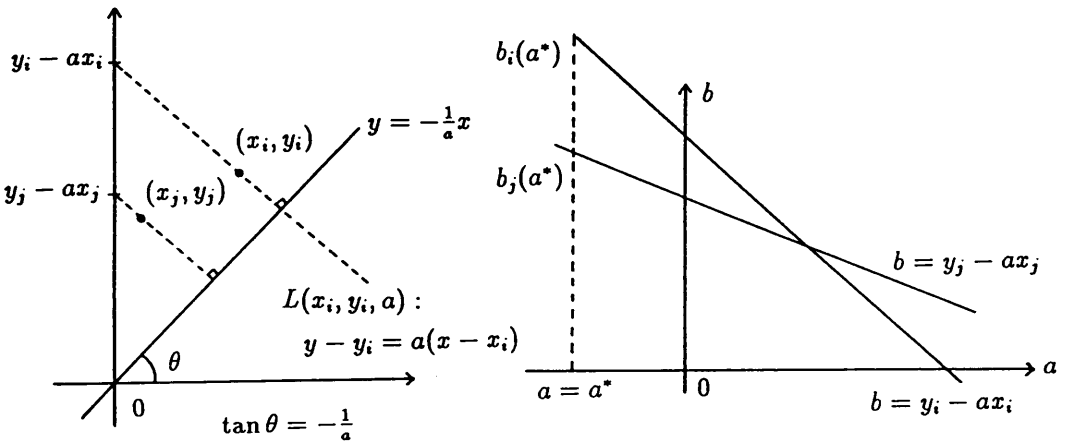


Fig. 1. Projection of points onto a line.



(a) Points in the xy-plane. **(b) Dual plane.**

Fig. 2. Dual transform.

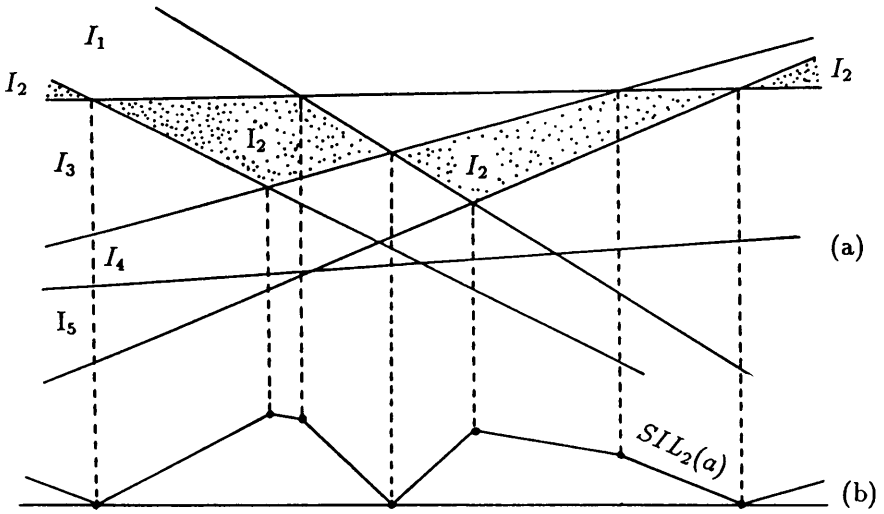


Fig. 3. Walls, intervals and silhouette.

(a) Small numbers represent walls and $I_1, I_2, \ldots, I_s$ are intervals.

(b) The silhouette for the interval I_2.