

PAPER

Inserting Points Uniformly at Every Instance

Sachio TERAMOTO[†], Tetsuo ASANO[†], Naoki KATOH^{††}, and Benjamin DOERR^{†††},

SUMMARY Arranging n points as uniformly as possible is a frequently occurring problem. It is equivalent to packing n equal and non-overlapping circles in a unit square. In this paper we generalize this problem in such a way that points are inserted one by one with uniformity preserved at every instance. Our criterion for uniformity is to minimize the gap ratio (which is the maximum gap over the minimum gap) at every point insertion. We present a linear time algorithm for finding an optimal n -point sequence with the maximum gap ratio bounded by $2^{\lfloor n/2 \rfloor / (\lfloor n/2 \rfloor + 1)}$ in the 1-dimensional case. We describe how hard the same problem is for a point set in the plane and propose a local search heuristics for finding a good solution.

key words: algorithm, circle packing, computational geometry, discrepancy, local search, uniformity

1. Introduction

The circle packing problem is to place n equal and non-overlapping circles in a unit square. It is one of the most important geometric optimization problems with a number of applications and has been intensively investigated [6], [12]–[14]. It is well known that the circle packing problem is equivalent to arranging n points in a unit square in such a way that the minimum pairwise distance is maximized. This problem seems to be computationally hard. In fact, no optimal solution is known for relatively large values of n , say $n > 100$.

The problem considered in this paper is a generalization of the point arranging problem, namely its online version. We want to insert n points one by one in such a way that uniformity is achieved at every insertion of a point. Since the solutions for the offline point arranging problems are different for different values of n , it would be impossible to derive good point sequences from such optimal solutions even if they were available. It should also be noted that a subsequence of an optimal point sequence is not optimal. Therefore, we cannot hope for an incremental algorithm constructing optimal point sequences.

It is not straightforward to define uniformity of points. The minimum pairwise distance is not good to measure the uniformity of points as it does not reflect large empty areas. We could also borrow a measure from discrepancy theory [5], [9]. Here we take a simple geometric shape R and count how many points are contained in R while moving R all over the unit cube. The uniformity is measured by the difference

between the largest and smallest counts for all possible sizes of the shape. A serious disadvantage of the measure is computational hardness. Also, it is not clear what shape R to use.

We define uniformity of point distribution using not only closest point pairs but also largest empty circles. Our criterion is to minimize the gap ratio, which is the maximum gap (diameter of a largest empty circle) over the minimum gap (the minimum pairwise distance). Note that this definition is extendability to higher dimensions since those gaps can be defined in any dimension.

This problem is closely related to an industrial application on digital halftoning, which is a technique to convert continuous-tone images into binary images for printing. One of the most popular methods for halftoning is Dithering, which binarizes an images using a threshold matrix called the dither matrix. The quality of output images heavily depends on this matrix. A target is an $n \times n$ matrix containing integers from 0 through $n^2 - 1$ in such a way that elements up to i are uniformly distributed for each $i = 1, 2, \dots, n^2 - 1$. Such a matrix is similar to the dither matrix called the blue-noise mask [11], [18]. Combinatorial approaches are also found for the problem, see e.g., [1]–[4], [7], [8], [17].

1.1 Problem Statement

Let $\mathbb{S}^d = [0, 1]^d$ be the unit cube in the d -dimensional space \mathbb{R}^d and $P = (p_1, \dots, p_n)$ be an n -point sequence contained in \mathbb{S}^d . We insert p_1, p_2, \dots, p_n in this order. For each $i = 1, 2, \dots, n$, we define a subsequence P_i of P by its first i points, i.e., $P_i = (p_1, \dots, p_i)$. With P_i , we associate a point set $S_i := \{p_1, \dots, p_i\} \cup S_0$, where S_0 is the set of the 2^d corner points of \mathbb{S}^d . The smallest among all pairwise distances in S_i is the minimum gap

$$g_i := \min_{p, q \in S_i, p \neq q} d(p, q),$$

where $d(p, q)$ is the Euclidean distance between two points p and q .

The maximum gap is defined via the largest empty circle. An empty circle is a circle whose center is located within the unit cube and contains no point of the set S_i . The diameter of the largest empty circle for the set S_i is the maximum gap G_i

$$G_i := \max_{p \in \mathbb{S}^d} \min_{q \in S_i} 2d(p, q)$$

Note that the point p in the definition above is an arbitrary

Manuscript received February 20, 2006.

Manuscript revised January 1, 2006.

Final manuscript received January 1, 2006.

[†]School of Information Science, JAIST, Japan

^{††}Graduate School of Kyoto University, Japan

^{†††}Max-Planck-Institut für Informatik, Saarbrücken, Germany

point in the unit cube. Now we define the i -th gap ratio by

$$r_i := G_i/g_i.$$

For a point sequence P , we define the maximum gap ratio as $R_P := \max_{i=1,\dots,n} r_i$. For a fixed integer n , we denote R_n the optimal gap ratio for any n -point sequence:

$$R_n := \min \{ R_P \mid P \text{ is an } n\text{-point sequence in } \mathbb{S}^d \}.$$

Given d and n , we want to find an n -point sequence P in \mathbb{S}^d that achieves the optimal (=minimal) gap ratio R_n . More formally, our problem is described as follows.

Problem 1:

Input: Integers d and n .

Ensured: Compute an optimal n -point sequence P in \mathbb{S}^d that achieves the optimal gap ratio R_n for n points.

1.2 Our contribution

We start with a simple greedy algorithm called incremental Voronoi insertion for points in the plane in Section 2. The Voronoi insertion generates a point sequence P with $R_P \leq 2$. It is also easy to extend this algorithm to higher dimensions while keeping the gap ratio 2.

In Section 3, we give a linear time algorithm that constructs an n -point sequence with maximum gap ratio bounded by $2^{\lfloor n/2 \rfloor / (\lfloor n/2 \rfloor + 1)}$ in the 1-dimensional unit cube \mathbb{S}^1 . We also show that the bound is optimal, which fully shows the 1-dimensional case.

Section 4 deals with the 2-dimensional case again. It looks quite challenging to find an optimal point sequence even for rather small values of n . Therefore, we give two simple heuristic algorithms finding a point sequence with maximum gap ratio smaller than that of the point sequence generated by the incremental Voronoi insertion. Since the Voronoi insertion gives an upper bound, our next goal is the following:

Problem 2:

Input: Integers d and n .

Output: An n -point sequence P in \mathbb{S}^d such that $R_P < 2$.

We have implemented our heuristic algorithm to find point sequences whose maximum gap ratios are strictly less than 2, which is achieved by the Voronoi insertion. Some such sequences are given together with related statistics on our experiments.

2. Simple Greedy Algorithm

We start with a simple greedy algorithm for inserting points uniformly. In this algorithm, we maintain a Voronoi diagram for a set of points which have already been inserted and its intersection with the boundary of the unit cube. Voronoi vertices and the intersections between Voronoi edges and cube surface are candidates for the next point to be inserted. We

evaluate each such vertex by the distance to its nearest point (site) and choose the one of the largest such distance as the next point to be inserted. This is why we call it incremental Voronoi insertion.

Define a point set $S_i^d = \{(x_1, x_2, \dots, x_d) \mid \text{exactly } i \text{ coordinates are either 0 or 1 and the remaining coordinates are } 1/2\}$ for $i < d$. For example, we have

$$S_0^3 = \{(1/2, 1/2, 1/2)\},$$

$$S_1^3 = \{(*, 1/2, 1/2), (1/2, *, 1/2), (1/2, 1/2, *)\},$$

$$S_2^3 = \{(*, *, 1/2), (*, 1/2, *), (1/2, *, *)\},$$

where $*$ indicates 0 or 1, that is, $(*, 1/2, 1/2)$ represents $(0, 1/2, 1/2), (1, 1/2, 1/2)$.

The first point to be inserted must be the unique element of S_0^d , i.e., $(1/2, 1/2, \dots, 1/2)$. Then, we insert points in the set S_1^d one by one, and continue to points in $S_2^d, S_3^d, \dots, S_{d-1}^d$. Suppose we have inserted all the points in $S_0^d, S_1^d, \dots, S_{d-2}^d$ and we are now going to insert $p_j = (0, 0, \dots, 0, 1/2)$, the first point in the set S_{d-1}^d . The point p_j is the mid-point of a cube edge by the definition. Thus, the minimum pairwise distance is $1/2$, that is, the minimum gap is $1/2$. Since this is the first point located on a cube edge, the empty ball centered at the next point $(0, 0, \dots, 0, 1, 1/2)$ that passes through the two points $(0, 0, \dots, 0, 0)$ and $(0, 0, \dots, 0, 1)$ remains empty. In fact, this ball is the largest empty ball. Its diameter is obviously 1. Therefore, the ratio is exactly 2 after the point.

We can also show that the maximum ratio before inserting this point is less than 2 and it remains so until the very last point of S_{d-1}^d . When we have inserted all the points in $S_0^d, S_1^d, \dots, S_{d-1}^d$, we can continue the same process again for 2^d sub-cubes in a recursive fashion. Thus, we can conclude that the above-mentioned approximation algorithm achieves the maximum ratio 2.

3. 1-dimensional problem

Our domain here is a unit interval $[0, 1]$. The two extremal points 0 and 1 are assumed to be placed in advance. We present a simple linear time strategy better than the incremental Voronoi insertion. Moreover, we show that the strategy is in fact optimal.

3.1 Lower bound on R_n

We first estimate the lower bound of R_n for an n -point sequence. Let $P = (p_1, p_2, \dots, p_n)$ be a finite sequence of n points in the unit interval $[0, 1]$ such that $p_i \neq p_j$ whenever $i \neq j$. For $i = 0, \dots, n$, the points p_1, \dots, p_i partition the unit interval into $i+1$ intervals of lengths $m_1^i, m_2^i, \dots, m_{i+1}^i$. Without loss of generality we may assume that $m_j^i \geq m_{j+1}^i$ for all i , $0 \leq i \leq n$ and $j, 1 \leq j \leq i$. Then, the maximum and minimum gaps are given by m_1^i and m_{i+1}^i , respectively. Hence, the ratio R_P for the sequence P is

$$R_P := \max_{1 \leq i \leq n} \frac{m_1^i}{m_{i+1}^i} \quad (1)$$

Put $M^i = \{m_1^i, \dots, m_{i+1}^i\}$ and regard it as a multi-set (i.e., it may contain elements more than once). Clearly, M^{i+1} is obtained from M^i by replacing one element from M^i by two which add up to the first one. The following lemma states that if $R_P \leq 2$, then this replaced element is always the largest.

Lemma 3.1: If $R_P \leq 2$, then for each $i = 0, \dots, n-1$ there are $a, b \in [0, 1]$ such that $m_1^i = a + b$ and $M^{i+1} = \{m_2^i, \dots, m_{i+1}^i, a, b\}$ (as multi-set) and one of a and b is a smallest element of M^{i+1} .

Proof Assume that $M^{i+1} = M^i \setminus \{m_j^i\} \cup \{a, b\}$ for some $j, 1 \leq j \leq i+1$ such that $m_j^i < m_1^i$ and $a + b = m_j^i$. W.l.o.g., let $b \leq a$. Then, $b \leq \frac{1}{2}m_j^i < \frac{1}{2}m_1^i$ and hence $R_P \geq m_1^{i+1}/b = m_1^i/b > 2$. If both a and b are greater than m_{i+1}^i , then again $R_P \geq m_1^i/m_{i+1}^i = (a+b)/m_{i+1}^i > 2m_{i+1}^i/m_{i+1}^i = 2$. \square

Note, however, that a priori we do not know that both a and b are not larger than m_{i+1}^i .

Lemma 3.2: Given an integer $n \geq 1$, the lower bound of R_n is $2^{\lfloor n/2 \rfloor / (\lfloor n/2 \rfloor + 1)}$.

Proof Assume that $R_P \leq 2$ for an n -point sequence. Let first n be even. Let $j, 1 \leq j \leq \frac{n}{2} + 1$ be such that $m_j^{n/2} \in M^n$. Such a j exists, since at most $n/2$ of the elements in $M^{n/2}$ are replaced in the sequel from $M^{n/2}$ to M^n . We have

$$\frac{m_1^{n/2}}{m_j^{n/2}} \leq \frac{m_1^{n/2}}{m_{n/2+1}^{n/2}} \leq R_P.$$

Also, for each $n/2 \leq i \leq n-1$, we have $R_P \geq m_1^{i+1}/m_{i+2}^{i+1} \geq m_1^{i+1}/m_{j/2}^{i+1}$ by Lemma 3.1. Since $m_j^{n/2} \in M^n$,

$$R_P \geq \frac{m_1^{n/2}}{m_j^{n/2}} \geq \frac{m_1^{n/2}}{m_1^n} = \prod_{i=n/2}^{n-1} \frac{m_1^i}{m_{i+1}^i} = \prod_{i=n/2}^{n-1} \frac{m_1^i}{m_{i+1}^i} \geq \left(\frac{2}{R_P}\right)^{n/2}.$$

We conclude $R_P \geq 2^{(n/2)/(n/2+1)}$. For n odd, let $P' = (p_1, \dots, p_{n-1})$. Then, $R_P \geq R_{P'}$ by definition and $R_{P'} \geq 2^{\lfloor n/2 \rfloor / (\lfloor n/2 \rfloor + 1)}$ by the above. So, $R_n \leq 2^{\lfloor n/2 \rfloor / (\lfloor n/2 \rfloor + 1)}$. This completes the proof of Lemma 3.2. \square

So, we have obtained the lower bound of R_n for n -point sequences. Now, what remains is to give an algorithm for computing an optimal point sequence P^* .

First, consider the following algorithm (Algorithm 1) suggested in the lower bound proof.

This strategy always puts a point p_i so that the gap ratio is equal to $2^{\lfloor n/2 \rfloor / (\lfloor n/2 \rfloor + 1)}$ for each i . If it is possible then the sequence obtained is optimal since its bound coincides with the lower bound. The strategy implicitly assumes that the smaller one of the new subintervals has the minimum length among current intervals. Unfortunately,

Algorithm 1: A naive strategy

Calculate $r = 2^{\lfloor n/2 \rfloor / (\lfloor n/2 \rfloor + 1)}$;

$p_1 = 1/(1+r)$;

for $i = 1$ **to** $n-2$ **do**

Let m_1^i and m_2^i be the current longest and second longest intervals, respectively;

Put a point p_{i+1} into m_1^i to partition it into two subintervals a and b so that $m_2^i / \min\{a, b\} = r$;

Put the last point p_n so as to partition the current longest interval into two intervals of the same lengths;

it is impossible to keep the ratio. The reason is as follows. Let a_i and b_i ($a_i > b_i$) be new subintervals resulting after the i -th insertion. Then, $M^1 = \{a_1, b_1\} = \{\frac{r}{r+1}, \frac{1}{r+1}\}$ and $M^2 = \{b_1, a_2, b_2\} = \{\frac{1}{r+1}, \frac{r-1}{r}, \frac{1}{r(r+1)}\}$. We insert p_3 into M^2 . Note that the maximum interval length in M^2 depends on the number of points to be inserted. If $b_1 \geq a_2$, (the case of $r \leq \frac{1+\sqrt{5}}{2}$), then $b_3 = \frac{a_2}{r} = \frac{r-1}{r^2}$ and $a_3 = \frac{1}{r+1} - b_3 = \frac{1}{r^2(r+1)}$. Since $a_3 < b_3$ for $r > \sqrt{2}$, $r_3 = a_2/a_3 = r(r^2-1) > R_n$. This suggests that if n is large enough, say $n > 3$, the assumption of above strategy does not hold. On the other hand, if $b_1 < a_2$ (the case of $r > \frac{1+\sqrt{5}}{2}$), then $r_2 = \frac{a_2}{b_2} = r^2 - 1$, and $2 < R_n^2 - 1$ for $n \geq 8$. Therefore, we cannot obtain an optimal point sequence P^* by the above strategy.

Observation 3.3: Gap ratios for the first $n-1$ points should be strictly less than $2^{\lfloor n/2 \rfloor / (\lfloor n/2 \rfloor + 1)}$, and moreover, these ratios are never determined until the last interval is fixed.

This Observation 3.3 suggests that an optimal point sequence of length n should be determined in a bottom-up fashion, that is, from the last interval to the first one.

3.2 An optimal point insertion strategy

A rough sketch of our strategy is as follows. Let (p_1, p_2, \dots, p_n) be a point sequence to be inserted in the unit interval $x_1 = [0, 1]$. We maintain all intervals generated during n insertions, and we denote by x_j the interval induced by the p_{j-1} . Hereafter, we denote the j -th interval by x_j , and unify x_j and its length $|x_j|$. Each point $p_i, i = 1, \dots, n$, is inserted into the current largest interval x_i to split it into two new subintervals x_{2i} and x_{2i+1} with $x_{2i} + x_{2i+1} = x_i$. An important observation here is that we can determine the point p_i so that it results in a sorted sequence $(x_{i+1}, x_{i+2}, \dots, x_{2i}, x_{2i+1})$ of intervals in the non-increasing order of their lengths. The process is terminated when the last point p_n is inserted to have a sequence $(x_{n+1}, x_{n+2}, \dots, x_{2n+1})$.

Now, let us describe how to determine the point sequence. It is divided into two subsequences at $k = \lfloor n/2 \rfloor$. For the first half (p_1, \dots, p_k) , the current longest interval x_i is unevenly partitioned into the new two subintervals x_{2i} and x_{2i+1} , so that $x_{2i} > x_{2i+1}$ and $x_i = x_{2i} + x_{2i+1}$. Since we are trying to achieve a ratio strictly less than 2, the ratio

x_{i+1}/x_{2i+1} must be strictly less than 2. For the remaining points (p_{k+1}, \dots, p_n) , the current longest interval x_i is partitioned evenly into two new subintervals x_{2j} and x_{2j+1} so that $x_{2j} = x_{2j+1} = x_j/2$ and x_{j+1}/x_{2j+1} is strictly less than 2, or equal to R_n . This is because the intervals x_{2i} and $x_{2i+1}, i = k+1, \dots, n$, will never be subdivided during the remaining insertion. Since minimum gaps are maximized by evenly partitioning, it minimizes the maximum gap ratios.

More concretely, we first compute the target ratio $R_n = 2^{k/(k+1)}$ where $k = \lfloor n/2 \rfloor$, and a magic number $y_1 = (2^{l-k} + 2 \sum_{i=2}^{k+1} \frac{R_n^{i-1}}{2^{i-1}})^{-1}$, where $l = \lceil n/2 \rceil$. Then, we fix the last $2k+2$ intervals;

$$\begin{aligned}
 x_{2l} &= x_{2l+1} = y_1 && \text{if } n \text{ is odd,} \\
 x_{2l+1} &= y_1 && \text{if } n \text{ is even,} \\
 x_{2(l+1)} &= x_{2(l+1)+1} = \frac{R_n}{2} y_1, \\
 x_{2(l+2)} &= x_{2(l+2)+1} = \left(\frac{R_n}{2}\right)^2 y_1, \\
 &\vdots \\
 x_{2(l+k)} &= x_{2(l+k)+1} = \left(\frac{R_n}{2}\right)^k y_1.
 \end{aligned}$$

The remaining intervals can be determined so that $x_i = x_{2i} + x_{2i+1}$, $i = k, k-1, \dots, 2, 1$. This strategy can be summarized in the following pseudo code.

Algorithm 2: An optimal strategy

input : An integer $n > 0$.
output: An optimal point sequence P , i.e., $R_P = R_n$.
 1 $R_n = 2^{\lfloor n/2 \rfloor / (\lfloor n/2 \rfloor + 1)}$;
 2 $y_1 = \left(2^{\lfloor n/2 \rfloor - \lfloor n/2 \rfloor} + 2 \sum_{i=2}^{\lfloor n/2 \rfloor + 1} \frac{R_n^{i-1}}{2^{i-1}}\right)^{-1}$;
 3 **if** n is odd **then** $x_{2\lfloor n/2 \rfloor} = x_{2\lfloor n/2 \rfloor + 1} = y_1$;
 4 **else** $x_{2\lfloor n/2 \rfloor + 1} = y_1$;
 5 **for** $i = 1$ **to** $\lfloor n/2 \rfloor$ **do** $x_{2(\lfloor n/2 \rfloor + i)} = x_{2(\lfloor n/2 \rfloor + i) + 1} = \left(\frac{R_n}{2}\right)^i \cdot y_1$;
 6 **for** $i = \lfloor n/2 \rfloor$ **downto** 1 **do** $x_i = x_{2i} + x_{2i+1}$;
 7 Compute a point sequence P from the interval sequence $(x_{i+1}, x_{i+2}, \dots, x_{2i}, x_{2i+1})$;

3.3 Configuration tree

Before showing the optimality and correctness of our strategy, we introduce a configuration tree to simplify the arguments for the proof. The tree describes how intervals are generated. Initially it consists of a root corresponding to the unit segment (or interval) x_1 . When an interval x_i is partitioned into two subintervals x_{2i} and x_{2i+1} , two corresponding nodes are created as children of the node for the interval x_i . Then, a set of internal nodes are those for x_1, x_2, \dots, x_i and the remaining nodes for x_{i+1}, \dots, x_{2i+1} are leaf nodes of the tree, which form an interval sequence $(x_{i+1}, \dots, x_{2i+1})$ in the order of their appearance. Fig. 1 shows an example of

the configuration tree for $n = 4$. The shaded nodes are leaf nodes. We can see how the intervals corresponding to leaf nodes subdivide the unit segment.

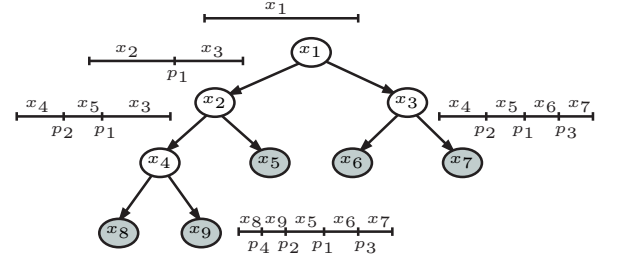


Fig. 1 Configuration tree for a point sequence $P = (p_1, p_2, p_3, p_4)$.

The above strategy constructs a configuration tree, and a partition of the unit segment is obtained. So, each interval length corresponding a leaf node is calculated using the magic number y_1 . Since each internal node has exactly two children and both intervals are known, all interval lengths are determined successively from bottom to top (root).

3.4 Optimality and correctness

Finally, we prove that the maximum gap ratio R_P of the point sequence P computed by our strategy is equal to R_n . The magic number y_1 plays a very important role to optimize R_P . Lemma 3.4 determines the value of y_1 and guarantees the optimality of the resulting point sequence. The correctness of the strategy is proved in Lemma 3.6.

Lemma 3.4: If any set of intervals $\{x_{i+1}, \dots, x_{2i+2}, x_{2i+3}\}$ are sorted in non-increasing order with respect to their lengths, then the above strategy achieves the maximum gap ratio $R_P = 2^{\lfloor n/2 \rfloor / (\lfloor n/2 \rfloor + 1)}$.

Proof Let y_i denote the length of $x_{2(l+i)+1}$ for $i = 0, 1, \dots, k$, where $k = \lfloor n/2 \rfloor$ and $l = \lceil n/2 \rceil$. Note that the node x_l has interval y_1 as one of the children in the tree configuration. Now, we assume the gap ratio r_i is defined by $\frac{x_{i+1}}{x_{2i+1}} = \frac{x_{i+1}}{y_i}$ for the $(l+i)$ -th insertion. By Lemma 3.6, this definition of r_i does not cause any inconsistency. From this fact, the minimum interval is y_i and the maximum interval is $x_{2(l+i)} + x_{2(l+i)+1} = 2y_{i+1}$, for $(l+i)$ -th insertion ($1 \leq i < k$). At the last insertion, the minimum interval is y_{k+1} and the maximum interval is y_1 . Therefore, the gap ratios r_i for $i = l, l+1, \dots, l+k$, are given as follow,

$$\begin{aligned}
 r_l &= \frac{x_{l+1}}{x_{2l+1}} = \frac{x_{2l+2} + x_{2l+3}}{x_{2l+1}} = \frac{2y_2}{y_1}, \\
 r_{l+1} &= \frac{x_{l+2}}{x_{2l+3}} = \frac{x_{2l+4} + x_{2l+5}}{x_{2l+3}} = \frac{2y_3}{y_2}, \\
 &\vdots \\
 r_{n-1} &= r_{l+k-1} = \frac{x_{l+k}}{x_{2(l+k-1)+1}} = \frac{x_{2(l+k)} + x_{2(l+k)+1}}{x_{2(l+k-1)+1}} = \frac{2y_{k+1}}{y_k}, \\
 r_n &= r_{l+k} = \frac{x_{l+k+1}}{x_{2(l+k)+1}} = \frac{y_1}{y_{k+1}}.
 \end{aligned}$$

Since $x_{2i+3} > x_{4i+2}$ for $i \leq l-1$, $r_i = \frac{x_{i+1}}{x_{2i+1}} = \frac{x_{2i+2}+x_{2i+3}}{x_{4i+2}+x_{4i+3}} \leq \frac{x_{2i+2}}{x_{4i+3}} = r_{2i+1}$. This implies $R_n = \max\{r_l, r_{l+1}, \dots, r_n\} \geq \max\{r_1, r_2, \dots, r_{l-1}\}$. Thus, R_n is minimized when

$$\begin{aligned} R_n &= (r_1 \cdot r_{l+1} \cdots r_{l+k})^{\frac{1}{k+1}} = \left(\frac{2y_2}{y_1} \frac{2y_3}{y_2} \cdots \frac{2y_{k+1}}{y_k} \frac{y_1}{y_{k+1}} \right)^{\frac{1}{k+1}} \\ &= 2^{\frac{k}{k+1}} = 2^{\lfloor \frac{n}{2} \rfloor / (\lfloor \frac{n}{2} \rfloor + 1)}. \end{aligned}$$

□

Since every $r_i = R_n$, we have $y_i = \frac{2}{R_n} y_{i+1}$ for $i = 1, \dots, k$, and $y_{k+1} = \frac{y_1}{R_n}$. Moreover, $y_1 = \left(\frac{2}{R_n}\right)^{i-1} y_{i+1}$ for $i = 2, \dots, k+1$. Thus, if y_1 is determined then so is every y_i . When n is odd, $1 = \sum_{i=n+1}^{2n+1} x_i = 2 \sum_{j=1}^{k+1} y_j = 2 \sum_{j=1}^{k+1} \left(\frac{R_n}{2}\right)^{j-1} y_1$ leads to $y_1 = \frac{1}{2 \sum_{j=1}^{k+1} \left(\frac{R_n}{2}\right)^{j-1}}$. Similarly, when n is even, $1 = \sum_{i=n+1}^{2n+1} x_i = y_1 + 2 \sum_{j=2}^{k+1} y_j$ gives $y_1 = \frac{1}{1 + 2 \sum_{j=2}^{k+1} \left(\frac{R_n}{2}\right)^{j-1}}$.

Observation 3.5: $y_1 \geq y_2 \geq \dots \geq y_{k+1}$.

The observation follows from the facts that $y_i = \frac{2}{R_n} y_{i+1}$ and $\frac{2}{R_n}$ is greater than 1.

To show the correctness of this strategy and the optimality of the sequence obtained, we have to prove that the sequence $(x_{i+1}, \dots, x_{2i}, x_{2i+1})$ generated by p_i is a sorted sequence in the non-increasing order of their lengths for every $1 \leq i \leq n$.

Lemma 3.6: Whenever our strategy partitions the interval x_i for every $1 \leq i \leq n$, the resulting intervals $x_{i+1}, x_{i+2}, \dots, x_{2i+1}$ are sorted in the non-increasing order, that is, we have $x_{i+1} \geq x_{i+2} \geq \dots \geq x_{2i} \geq x_{2i+1}$.

Proof Proof is by induction on the level of a tree configuration of size $2n+1$. The level of a node v is defined as $\lfloor \log(2n+1) \rfloor - \text{height of } v$. So, all leaf nodes may be in the level 0 or 1, and the level of root x_1 is $\lfloor \log(2n+1) \rfloor$, (see Fig. 2).

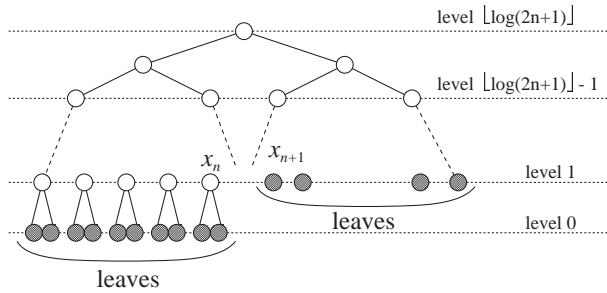


Fig. 2 The levels of tree corresponding the behavior of our strategy.

When $2^h = n+1$, where $h = \lfloor \log(2n+1) \rfloor$, all leaf nodes are in the level 0. In this case, from Observation 3.5, the statement $x_{n+1} \geq \dots \geq x_{2n+1}$ holds. When $n+1 \neq 2^h$, the intervals x_n and x_{n+1} are both in the same level 1. Then, we have

$$\begin{aligned} x_n &= 2y_{k+1} = 2 \left(\frac{R_n}{2} \right)^k y_1 = \frac{R_n^{k+1}}{R_n 2^{k-1}} y_1 = \frac{2^k}{R_n 2^{k-1}} y_1 \\ &= \frac{2}{R_n} y_1 > y_1 = x_{n+1}. \end{aligned}$$

On the remaining nodes in level 1, both children of a node are the intervals of the same length $2y_j$. Hence, for two intervals x_i and x_{i+1} , we have $x_i \geq x_{i+1}$ by Observation 3.5.

Let $I^i = (z_1^i, \dots, z_{2^i}^i)$ be the intervals in the level i , where z_1^i and $z_{2^i}^i$ are the leftmost and rightmost intervals in the level i , respectively. Now, we assume that the statement holds up to the level i , that is, $z_1^i \geq z_2^i \geq \dots \geq z_{2^i}^i \geq \dots \geq x_{2n+1}$. By the induction hypothesis, we have $z_{2^{i-1}}^{i-1} = z_{2^{i-1}}^i + z_{2^i}^i \geq z_1^{i+1} + z_2^{i+1} = z_1$. We also have $z_j^{i-1} \geq z_{j+1}^{i-1}$ by a similar argument. □

Thus, we have a conclusion on 1-dimensional dispersion problem.

Theorem 3.7: Given an integer n , our strategy gives an optimal solution with the maximum gap ratio being $2^{\frac{\lfloor n/2 \rfloor}{\lfloor n/2 \rfloor + 1}}$ on the 1-dimensional dispersion problem in $O(n)$ time.

4. 2-dimensional problem

4.1 Notations and analytical results

Let $s_1 = (0, 0)$, $s_2 = (1, 0)$, $s_3 = (1, 1)$, and $s_4 = (0, 1)$ be the four corner points of \mathbb{S}^2 . For each point set S_i after inserting i points in P , we define two empty circles C_i and c_i : The diameter of C_i is G_i and the center p of C_i satisfies $\min_{s \in S_i} d(s, p) = G_i/2$. The diameter of c_i is g_i and its center is the midpoint of the closest pair of points. Note that the two empty circles are not unique, since the maximum gap and the minimum one may be defined by some of the triples or pairs. We break ties arbitrarily to choose C_i and c_i . For any three different points p_1, p_2 , and p_3 , not on a line, let $C(p_1, p_2, p_3)$ be the circle passing through the three points. The interior of a circle C is denoted by $\text{int } C$ and the diameter of C is denoted by $\text{diam}(C)$. The gap ratio r_i is defined by $r_i = G_i/g_i = \text{diam}(C_i)/\text{diam}(c_i)$.

Lemma 4.1: For $i = 1, 2, \dots, n-1$, if $\max_{1 \leq j \leq i} r_j < 2$, then p_{i+1} must be inserted in $\text{int } C_i$, to keep $r_{i+1} < 2$.

Proof If p_{i+1} does not lie in $\text{int } C_i$, then $G_{i+1} = G_i$. We have $g_{i+1} \leq G_i/2$ since there is no empty circle whose diameter is greater than that of the largest empty circle. Hence $r_{i+1} \geq 2$. □

Fact 4.2: For two acute triangles $\triangle ABC$ and $\triangle DEF$, if $\angle ABC \leq \angle DEF$ and $\angle BCA \geq \angle EFD$, then

$$\frac{|AB|}{|CA|} \geq \frac{|DE|}{|FD|}.$$

We have equality if $\angle ABC = \angle DEF$ and $\angle BCA = \angle EFD$. See Fig. 3 for an example.

Lemma 4.3: The last point p_n must lie at the center of C_{n-1} to minimize the maximum gap ratio.

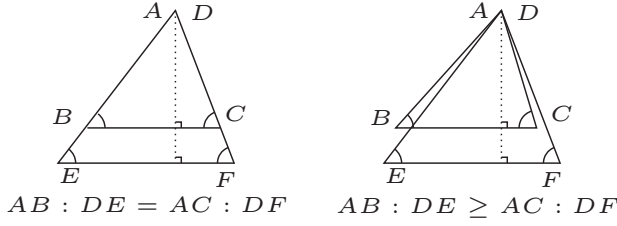


Fig.3 Illustration for Fact 4.2

Proof By Lemma 4.1, we assume that p_n is inserted into the interior of C_{n-1} . If C_{n-1} is not unique, then this lemma immediately holds, since it must maximize the minimum gap g_n .

We assume that C_{n-1} passes through three points a, b and c in the counter-clockwise order. Let C' be the other empty circle (not C_{n-1}) passing through a and b .

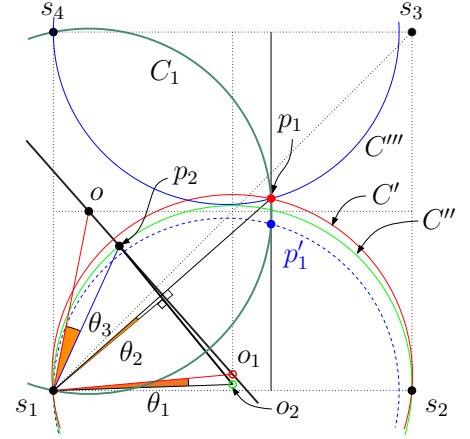
We move p_n along the perpendicular bisector of a and b so as to decrease g_n . Note that we may also have to consider motions between b and c , and between c and a . But similar arguments can be appropriately applied.

In this situation, a pair of points defining g_n is never changed, i.e., $g_n = d(p_n, a) = d(p_n, b)$. However, a triple (or pair) of points defining G_n may change. There are two kinds of meaningful circles which may define G_n ; the first one is C' defined above and the second one is the empty circle C'' that passes through a and p_n . The other circles may lead to $r_n \geq 2$, or may not lead to a better r_n than that of C' or C'' . Note that p_n has to be inserted at the center of C_{n-1} to maximize g_n , when $C_n = C'$. On the other hand, when $C_n \neq C'$, p_n lies in $\text{int } C'$.

Now, we assume $C_n = C''$. Let o be the center of C_{n-1} , o_1 be the center of C' and o_2 be the center of C'' . Consider two triangles $\Delta_1 = \Delta(a, o_2, p_n)$ and $\Delta_2 = \Delta(a, o_1, o)$. Since p_n is in C' , it can be seen that $\angle a o_2 p_n < \angle a o_1 o$ and $\angle a p_n o_2 > \angle a o o_1$ by simple calculations. Hence, we have $d(a, o_2)/d(a, p_n) > d(a, o_1)/d(a, o)$ from Fact 4.2. This concludes the proof. \square

By Lemma 4.3, we can find an optimal 2-point sequence. Fig.4 shows the notations for an instance of $n = 2$. In this figure, we consider that p_1 does not lie on the line $y = \frac{1}{2}$. The three circles C', C'' and C''' are shown, where C''' is the largest empty circle passing through p_1 when p_2 is in the largest empty circle passing through the symmetric point p'_1 of p_1 with respect to $y = \frac{1}{2}$, but the meaningful circles are just C' and C'' . Since $\theta_1 - \theta_2 > 0$ and $\theta_3 - \theta_2$, p_2 is put at the center of C_1 , from Fact 4.2 and Lemma 4.3.

We can assume that p_1 lies on the line $y = \frac{1}{2}$, to maximize g_1 . Since we can specify p_2 once p_1 is determined, we only examine the x -coordinate of p_1 . The maximum gap ratio R_p is minimized when $g_1 = g_2$, and then an optimal point pair satisfies $G_1^2 = 2g_1G_2$, by simple observations. Hence, for example, these gaps G_1, g_1 and G_2 are given by


 Fig.4 Notations of Lemma 4.3 for an instance of $n = 2$.

$$G_1 = \frac{4x_1^2 - 8x_1 + 5}{4 - 4x_1}, \quad g_1 = \sqrt{x_1^2 + \frac{1}{4}}, \quad \text{and}$$

$$G_2 = 2\sqrt{\frac{1}{4} + \left(x_1 - \frac{1}{2}\right)^4}.$$

Solving this simultaneous equations, we obtain the coordinates of optimal points;

$$p_1^* = (0.273704, 0.5), \quad \text{and}$$

$$p_2^* = (0.808958, 0.5).$$

Next, we consider the cases of $n = 3$ and larger n . They are more complicated and may not be solvable in an analytical sense.

Lemma 4.4: If $n \geq 3$ and the first point p_1 lies on the line $y = \frac{1}{2}$ or the line $x = \frac{1}{2}$, then the maximum gap ratio is greater than or equal to 2.

Proof We assume that the first point p_1 lies on the line $y = \frac{1}{2}$. When p_2 is inserted into $\text{int } C_1 \cap \text{int } C(p_1, s_1, s_2)$, p_3 should be inserted at the center of $C(p_1, s_3, s_4)$ from Lemma 4.3, and then C_3 is defined by $C(p_2, s_1, s_2)$. Since $\text{diam}(C_3) \leq 1$ and $\text{diam}(C(p_1, s_3, s_4)) \geq 1$, we have $r_3 \geq 2$. So, p_2 has to be inserted anywhere in $\text{int } C_1 \setminus \text{int } C(p_1, s_1, s_2) \cap \mathbb{S}^2$. Hence, C_2 is defined as the circle passing through p_1, s_1 and s_2 . By Lemma 4.1, p_3 is inserted in the interior of C_2 . Therefore, the third gap ratio r_3 is at least 2, since $\text{diam}(C_2) = \text{diam}(C_3)$ and $\text{diam}(c_2) \leq \frac{1}{2}\text{diam}(C_2)$.

By the symmetry, a similar argument can be applied when p_1 lies on the line $x = \frac{1}{2}$. \square

Lemma 4.5: When $n = 3$, the second point p_2 should be inserted at the center of C_1 .

Proof Let C'_1 and C''_1 be the second and third largest empty circles of S_1 . Consider the case in which there exist exactly two circles, C_1 and C'_1 , with their diameters greater than 1, at the end of the first insertion. Then, p_2 must be in $\text{int } C'_1 \cap \{p \in S \mid d(p_1, p) > \frac{1}{2}\} \cap \{p \in S \mid d(s_i, p) > \frac{1}{2}\}$, where s_i is the nearest corner point of S , since C_2 passes through p_2 , $\text{diam}(C_2) > 1$ and $g_2 = \max\{d(p_1, p_2), d(s_i, p_2)\} < \frac{1}{2}$. If such

intersection does not exist, then we can see $R_P \geq 2$. Let x and x' be the centers of C_1 and C'_1 , respectively. Since p_2 is inserted in that intersection, $C_2 = C(p_2, s_i, s_j)$, where s_j is the second nearest corner point of \mathbb{S}^2 from p_2 . Let x'' be the center of C_2 . Now, consider two triangles, $\Delta s_i x'' p_2$ and $\Delta s_i x' x$. From Fact 4.2, p_2 is inserted at the center of C_1 , to minimize r_2 .

Next, consider the next case that there are three empty circles, C_1, C'_1 , and C''_1 , with diameters greater than 1, at the end of the first insertion. This case occurs when p_1 is contained in exactly one circle $C(o, s_i, s_j)$, where o is the center $(\frac{1}{2}, \frac{1}{2})$, and s_i and s_j are corner points of S . We assume that p_2 is inserted in $\text{int } C'_1 \cap \{p \in S \mid d(p_1, p) > \frac{1}{2} \text{diam}(C''_1)\} \cap \{p \in S \mid d(s_i, p) > \frac{1}{2} \text{diam}(C''_1)\}$, where s_i is the nearest corner point of S from p_2 , the maximum gap ratio may be less than 2. If the intersection does not exist, then $R_P \geq 2$. However, the same argument as above applies. Therefore, p_2 should be inserted at the center of C_1 , if $n = 3$. \square

Lemma 4.6: When $n = 3$, all largest empty circles C_1, C_2 and C_3 pass through p_1 .

Proof It is obvious that C_1 and C_2 pass through p_1 from Lemmas 4.4 and 4.5. If p_2 lies on the line $y = \frac{1}{2}$, then p_3 lies on the line $x = \frac{1}{2}$, and vice versa. We can assume that p_2 and p_3 are located on the boundary of the cube $[0, \frac{1}{2}] \times [0, \frac{1}{2}]$ from Lemmas 3.6 and 4.5. Then, p_1 lies in the (open) square $(\frac{1}{2}, 1) \times (\frac{1}{2}, 1)$. Hence, C_3 passes through p_1 , since the open half plane $y > \frac{1}{2}$ contains p_1 but not p_2 or p_3 . \square

Since p_2 and p_3 are inserted at $\text{center}(C_1)$ and $\text{center}(C_2)$, respectively, we obtain $g_2 = \frac{1}{2}G_1$ and $g_3 = \frac{1}{2}G_2$. In order to minimize R_P , we take geometric average among r_1, r_2 and r_3 ;

$$\begin{aligned} R_P &= \sqrt[3]{r_1 \cdot r_2 \cdot r_3} = \sqrt[3]{\frac{G_1}{g_1} \frac{G_2}{g_2} \frac{G_3}{g_3}} = \sqrt[3]{\frac{G_1}{g_1} \frac{G_2}{\frac{1}{2}G_1} \frac{G_3}{\frac{1}{2}G_2}} \\ &= \sqrt[3]{2^2 \frac{G_3}{g_1}}. \end{aligned} \quad (2)$$

Hence, if we can show $\frac{G_3}{g_1} < 2$, then $R_P < 2$ is obtained when $n = 3$. The problem is to find a point p_1 which minimizes the value of equation (2). We can formulate it as a non-linear programming problem. However, it seems to be difficult to specify an optimal position of p_1 satisfying the above conditions, and analytically solving the exact positions in an optimal point sequence is too complicated even if n is rather small, say $n = 3$. So, we propose a heuristic algorithm for finding a good point sequence.

4.2 Heuristic Algorithms

We present a simple heuristic algorithm based on local search. First, we describe a procedure to compute the maximum gap ratio, for a given n -point sequence P . Then, we show a main procedure which treats n -point sequence

(p_1, \dots, p_n) as a point $(x_1, y_1, x_2, y_2, \dots, x_n, y_n)$ in the $2n$ -dimensional space \mathbb{R}^{2n} to find a best point by examining its neighborhood in \mathbb{R}^{2n} . This technique is similar to the lifting technique common in computational geometry.

Algorithm 3: ComputeMaxGapRatio(P)

```

input : A point sequence  $P = (p_1, p_2, \dots, p_n)$ 
output: The maximum gap ratio  $R_P$ 
1 Let  $S_0$  be the corner points of  $\mathbb{S}^2$ ;
2  $S \leftarrow S_0$ ;  $r \leftarrow 0$ ;
3 for  $i = 1, \dots, n$  do
4    $S \leftarrow S \cup \{p_i\}$ ;
5   Compute the maximum gap  $G_i$  and the minimum gap  $g_i$ ;
6   if  $r < G_i/g_i$  then  $r \leftarrow G_i/g_i$ ;
7 return  $r$ ;
```

Algorithm 3 computes the maximum gap ratio for a given n -point sequence. It runs in $O(n^2)$ time. In particular, we maintain a planar subdivision by a Delaunay triangulation [15] for each S_i . The planar subdivision by Delaunay triangulation is a planar graph. So, each face contributes to an empty circle and each edge represents the neighborhood relation between two connecting vertices (see [16]). Hence, we obtain the gaps G_i and g_i in linear time, since the reconstruction of the subdivision is the crucial part.

Algorithm 4 is a main procedure of our heuristics. Given three parameters n, m and k , the algorithm iterates local search k times starting from a randomly generated point sequence. In each iteration we compute a local optima of an n -point sequence. The parameter m is used to specify a termination condition to guarantee the accuracy of solutions obtained.

Algorithm 4: A simple local search heuristic algorithm

```

input : Integers  $n, m$ , and  $k$ .
output: A good  $n$ -point sequence.
1 Let  $e_1$  and  $e_2$  be the base unit vectors;
2  $r_{opt} \leftarrow \infty$ ; threshold  $\leftarrow 2^{-m}$ ;
3 for  $i = 1$  to  $k$  do
4   Initialize  $P$  by a randomly generated  $n$ -point sequence;
5    $\varepsilon \leftarrow \frac{1}{2}$ ;
6    $r_{min} \leftarrow \text{ComputeMaxGapRatio}(P)$ ;
7   repeat
8     foreach  $p'_1 \in \{p_1 \pm \varepsilon e_1, p_1 \pm \varepsilon e_2\}$  do
9       foreach  $p'_2 \in \{p_2 \pm \varepsilon e_1, p_2 \pm \varepsilon e_2\}$  do
10         $\vdots$ 
11        foreach  $p'_n \in \{p_n \pm \varepsilon e_1, p_n \pm \varepsilon e_2\}$  do
12           $P' \leftarrow (p'_1, p'_2, \dots, p'_n)$ ;
13           $r \leftarrow \text{ComputeMaxGapRatio}(P')$ ;
14          if  $r < r_{min}$  then  $r_{min} \leftarrow r$ ;  $P \leftarrow P'$ ;
15        if  $r_{min}$  is updated then  $P'' \leftarrow P$ ;
16        else  $\varepsilon \leftarrow \frac{1}{2}\varepsilon$ ;
17      until  $\varepsilon < \text{threshold}$ ;
18    if  $r_{opt} > r_{min}$  then  $r_{opt} \leftarrow r_{min}$ ;  $P^* \leftarrow P''$ ;
19 return  $P^*$ ;
```

4.3 Experimental Results

We have implemented Algorithm 4 to evaluate the accuracy of the solutions obtained by the heuristic algorithm. Table 1 describes our environment of the experiment. We designed the algorithm using the exact computation in LEDA[10] for the sake of accuracy and for robustness.

Table 1 The environment of experiment

Workstation	CPU
Dell PowerEdge SC1425 Server	Intel® Xeon™ 3.6GHz
Main memory	OS
8GB	RedHat Enterprise Linux 3
Compiler	External library
g++-3.4.2	LEDA-5.0.1

Table 2 shows the best R_p values obtained by Algorithm 4. For each $n = 2, 3, 4, 5$, we executed the algorithm more than 1000 times with the threshold less than 10^{-8} . For each of $n = 6, 7, 8$, we executed it with 500, 100, and 20 trials with the same accuracy.

Table 2 The best solutions returned by Algorithm 4

n	2	3	4	5
R_p	1.87804	1.92716	1.927164	1.92716
n	6	7	8	9
R_p	1.927203	1.99312	2.008371	–

As mentioned above, we have an exact value of R_2 and a property for R_3 . The computed value R_2 shown in the table finds to be close enough to the exact value of R_2 . The 3-point sequence achieving the computed value of R_3 shown in the table satisfies the property $\frac{G_3}{g_1} < 2$, and we conjecture that the point sequence is optimal. In addition to this, the obtained point sequences for $n = 4, 5, 6$ may also be optimal, since these maximum gap ratios are roughly the same.

There is a gap between the results for $n = 6$ and $n = 7$. We have obtained a better sequence than that of Voronoi insertion. However, in the case of $n = 8$, we did not obtain a sequence with the maximum gap ratio less than 2. In our environment of experiment, we gave up to apply the algorithm for $n \geq 8$, since it is too slow. In fact, it took one day per one trial.

We could use those point sequences obtained above as seed point sequences and perform the incremental Voronoi insertion. This is our second heuristic algorithm.

We have implemented the above-stated strategy using the 7-point sequence shown in Table 3 as a starting seed point sequence. The initial maximum gap ratio is 1.993124. Fig.5 indicates the resulting point distribution. The maximum gap ratio of this point sequence is actually 1.99921.

Furthermore, we consider the irregularity of the final point distribution for each of our results and Voronoi insertion. In order to enhance the difference between them, we

Table 3 A good seed point sequence and the initial maximum gap ratio.

p_1	p_2
(0.769146, 0.501913)	(0.263398, 0.508807)
p_3	p_4
(0.499994, 0.0637435)	(0.477718, 0.891089)
p_5	p_6
(2.0687e-05, 0.317322)	(8.21674e-06, 0.662797)
p_7	R_p
(0.999993, 0.304037)	1.993124

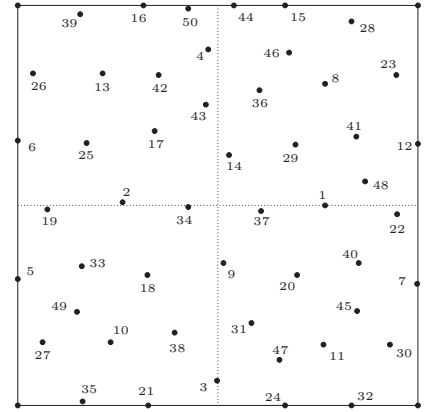


Fig. 5 A good 50-point sequence with the maximum gap ratio bounded by 1.99921.

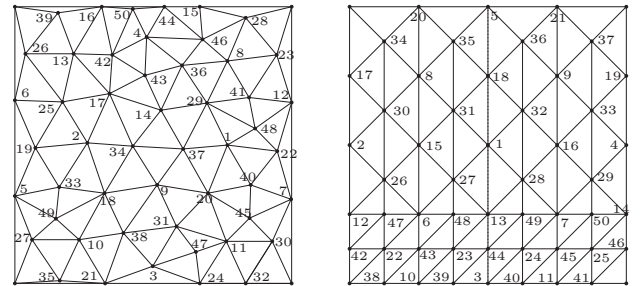


Fig. 6 The Delaunay triangulations for the resulting point distributions. In the triangulation for our 50-point sequence the maximum gap ratio is bounded by 1.99921, which is shown in the left. The triangulation for the incremental Voronoi insertion is given to the right.

use a Delaunay triangulation shown in Fig. 6. Our distribution is pretty irregular, compared with that obtained by the Voronoi insertion.

One of the notable remarks is that Voronoi insertion easily gives a uniform point sequence in our criterion, but the final distribution is globally non-uniform and locally regular.

5. Conclusions

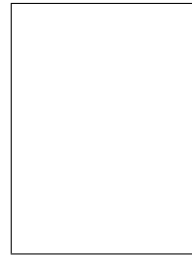
In this paper we have presented a preliminary result on dispersion. One of the most important future works is to extend the result to higher dimensions. We showed some results on lower and upper bounds of the maximum gap ratio for the planar case, but none in the higher dimensions.

Acknowledgments

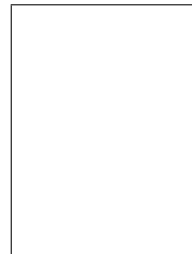
The first two authors would like to thank Ryuhei Uehara and Taisuke Shimamoto for reading the first version of the manuscript. This work was partially supported by the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for Scientific Research on Priority Areas and Scientific Research (B).

References

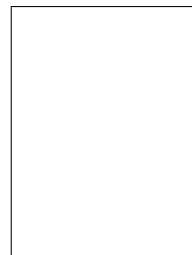
- [1] B. Aronov, T. Asano, Y. Kikuchi, S. C. Nandy, S. Sasahara, and T. Uno, "A Generalization of Magic Squares with Applications to Digital Halftoning," to appear in *Theory of Computing System*.
- [2] T. Asano, "Computational Geometric and Combinatorial Approaches to Digital Halftoning," *Proc. of Conferences in Research and Practice in Information Technology*, 51, p.3, 2006.
- [3] T. Asano, P. Brass, and S. Sasahara, "Disc Covering Problem with Application to Digital Halftoning," *Proc. Int. Conf. on Computer Science and Applications*, vol.3, pp.11–21, 2004.
- [4] T. Asano, N. Katoh, K. Obokata, and T. Tokuyama, "Matrix rounding under the L_p -discrepancy measure and its application to digital halftoning," *SIAM J. Computing*, 32, 6, pp.1423–1435, 2003.
- [5] B. Chazelle: "The Discrepancy Method: Randomness and Complexity," Cambridge University Press, 2000.
- [6] C. R. Collins and K. Stephenson, "A Circle Packing Algorithm," *Comput. Geom., Theory and Applications*, 25, 3, pp.233–256, 2003.
- [7] B. Doerr, "Nonindependent Randomized Rounding and an Application to Digital Halftoning," *SIAM Journal on Computing*, 34, 2, pp.299–317, 2005.
- [8] S. Gooran, "Context Dependent Colour Halftoning in Digital Printing," Department of Science and Technology, Linköping University, Sweden Proceedings of the Conference on Image Processing, Image Quality, Image Capture Systems (PICS-00), pp.242–246, 2000.
- [9] J. Matoušek: "Geometric Discrepancy," Springer, 1999.
- [10] K. Mehlhorn and S. Näher: "LEDA – A Platform of Combinatorial and Geometric Computing," Cambridge University Press, Cambridge, England, 1999.
- [11] T. Mitsa and K. J. Parker, "Digital Halftoning using a Blue-Noise Mask," *Journal of the Optical Society of America A*, 9, 11, pp.1920–1929, 1992.
- [12] K. J. Nurmela, P. R. J. Östergård and R. aus dem Spring, "Asymptotic Behavior of Optimal Circle Packings in a Square," *Canadian Mathematical Bulletin*, 42, 3, pp.380–385, 1999.
- [13] K. J. Nurmela and P. R. J. Östergård, "Packing up to 50 Equal Circles in a Square," *Discrete and Computational Geometry*, 18, 1, pp.111–120, 1997.
- [14] K. J. Nurmela, "More Optimal Packings of Equal Circles in a Square," *Discrete and Computational Geometry*, 22, 3, pp.439–457, 1999.
- [15] A. Okabe, B. Boots, K. Sugihara, and S.N. Chiu: "Spatial Tessellations: Concepts and Applications of Voronoi Diagrams," John Wiley & Sons, 2000.
- [16] F.P. Preparata and M.I. Shamos: "Computational Geometry: An Introduction," Springer-Verlag, 1985.
- [17] K. Sadakane, N. Takki Chebihi, and T. Tokuyama, "Discrepancy-based digital halftoning: Automatic evaluation and optimization," *Interdisciplinary Information Sciences*, 8, 2, pp.219–234, 2002.
- [18] R. Ulichney, "Digital Halftoning," MIT Press, 1987.



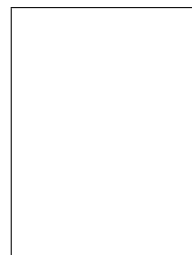
Sachio Teramoto received M.S. degree from School of Information Science, Japan Advanced Institute of Science and Technology (JAIST) in 2003. He is presently a Ph.D student under the supervision of Prof. Asano. His research interests include design and analysis of combinatorial and geometric algorithms.



Dr. Tetsuo Asano was born in Kyoto Prefecture, Japan, in 1949. He got B.E., M.E., and Ph.D degrees from Osaka University, Japan, in 1972, 1974, and 1977, respectively. In 1977 he joined Osaka Electro-Communication University as a lecturer and moved to JAIST (Japan Advanced Institute of Science and technology) in 1997. He is now a professor in School of Information Science. His research interest includes algorithms and data structures, especially in computational geometry, combinatorial optimization, computer graphics, computer vision using geometric information, and VLSI layout design. He has been serving as editors of several journals including *Discrete and Computational Geometry*, *Computational Geometry: Theory and Applications*, *International Journal of Computational Geometry and Applications*, and *Theory of Computing Systems*. He served as a chair of a Special Interest Group on Algorithms of Information Processing Society of Japan in 1994–1996. He is fellows of Association of Computing Machinery (2001) and Information Processing Society of Japan (2004).



Naoki Katoh received the B.Eng. M.Eng. and Dr.Eng. degrees in Applied Mathematics and Physics from Kyoto University in 1973, 1975 and 1981, respectively. He was an Assistant Professor during 1981–1982, an Association Professor during 1982–1990, and a Professor during 1990–1997, respectively, at Department of Management Science of Kobe University of Commerce. In 1997 he joined Kyoto University where he is currently a Professor at Department of Architecture and Architectural Engineering. His research interests include the design and analysis of combinatorial and geometric algorithms, data mining and architectural information systems. He is a member of IPSJ, OR Soc. Japan, Japan SIAM and ACM.



Dr. habil. Benjamin Doerr was born in Munich, Germany, in 1971. He received diploma, PhD and habilitation in mathematics from Christian-Albrechts-Universität zu Kiel, Germany, in 1998, 2000 and 2005, respectively. He was a research assistant in Kiel in 2001–2005. In 2005, he joined the Max-Planck-Institut für Informatik in Saarbrücken, Germany, as a senior researcher. His research interests include discrete and probabilistic mathematics and algorithmics, in particular, discrepancy theory, randomized rounding and evolutionary algorithms.