A Randomized Algorithm for Online Unit Clustering*

Timothy M. Chan and Hamid Zarrabi-Zadeh

School of Computer Science, University of Waterloo
Waterloo, Ontario, Canada, N2L 3G1
{tmchan, hzarrabi}@uwaterloo.ca

Abstract. In this paper, we consider the online version of the following problem: partition a set of input points into subsets, each enclosable by a unit ball, so as to minimize the number of subsets used. In the one-dimensional case, we show that surprisingly the naïve upper bound of 2 on the competitive ratio can be beaten: we present a new randomized 15/8-competitive online algorithm. We also provide some lower bounds and an extension to higher dimensions.

1 Introduction

Clustering problems—dividing a set of points into groups to optimize various objective functions—are fundamental and arise in a wide variety of applications such as information retrieval, data mining, and facility location. We mention two of the most basic and popular versions of clustering:

Problem 1 (k-Center) *Given a set of n points and a parameter k, cover the set by k congruent balls, so as to minimize the radius of the balls.*

Problem 2 (Unit Covering) *Given a set of n points, cover the set by balls of unit radius, so as to minimize the number of balls used.*

Both problems are NP-hard in the Euclidean plane [10, 19]. In fact, it is NP-hard to approximate the two-dimensional k-center problem to within a factor smaller than 2 [9]. Factor-2 algorithms are known for the k-center problem [9, 11] in any dimension, while polynomial-time approximation schemes are known for the unit covering problem [14] in fixed dimensions.

Recently, many researchers have considered clustering problems in more practical settings, for example, in the online and data stream models [4, 5, 12], where the input is given as a sequence of points over time. In the online model, the solution must be constructed as points arrive and decisions made cannot be subsequently revoked; for example, in the unit covering problem, after a ball is opened to cover an incoming point, the ball cannot be removed later. In the related streaming model, the main concern is the amount of working space; as

* Work of the first author has been supported in part by NSERC.

points arrive, we must decide which point should be kept in memory. We focus on the online setting in this paper.

The online version of the unit covering problem is one of the problems addressed in the paper by Charikar et al. [4]. They have given an upper bound of $O(2^d d \log d)$ and a lower bound of $\Omega(\frac{\log d}{\log \log \log d})$ on the competitive ratio of deterministic online algorithms in d dimensions; for $d = 1$ and 2, the lower bounds are 2 and 4 respectively.

In this paper, we address the online version of the following variant:

Problem 3 (Unit Clustering) *Given a set of n points, partition the set into clusters (subsets), each of radius at most one, so as to minimize the number of clusters used. Here, the* radius *of a cluster refers to the radius of its smallest enclosing ball.*

At first glance, Problem 3 might look eerily similar to Problem 2; in fact, in the usual offline setting, they are identical. However, in the on-line setting, there is one important difference: as a point p arrives, the unit clustering problem only requires us to decide on the choice of the cluster containing p, not the ball covering the cluster; the point cannot subsequently be reassigned to another cluster, but the position of the ball may be shifted.

We show that it is possible to get better results for Problem 3 than Problem 2. Interestingly we show that even in one dimension, the unit clustering problem admits a nontrivial algorithm with competitive ratio better than 2, albeit by using randomization. In contrast, such a result is not possible for unit covering. To be precise, we present an online algorithm for one-dimensional unit clustering that achieves expected competitive ratio 15/8 against oblivious adversaries. Our algorithm is not complicated but does require a combination of ideas and a careful case analysis. We contrast the result with a lower bound of 4/3 and also extend our algorithm for the problem in higher dimensions under the L_∞ metric.

We believe that the one-dimensional unit clustering problem itself is theoretically appealing because of its utter simplicity and its connection to well-known problems. For example, in the exact offline setting, one-dimensional unit clustering/covering is known to be equivalent to the dual problem of finding a largest subset of disjoint intervals among a given set of unit intervals—i.e., finding maximum independent sets in unit interval graphs. Higher-dimensional generalizations of this dual independent set problem have been explored in the map labeling and computational geometry literature [2, 3, 8], and online algorithms for various problems about geometric intersection graphs have been considered (such as [18]). The one-dimensional independent set problem can also be viewed as a simple scheduling problem (dubbed "activity selection" by Cormen et al. [6]), and various online algorithms about intervals and interval graphs (such as [1, 7, 16, 17]) have been addressed in the literature on scheduling and resource allocation. In the online setting, one-dimensional unit clustering is equivalent to clique partitioning in unit interval graphs, and thus, equivalent to coloring in unit co-interval graphs. It is known that general co-interval graphs can be colored with competitive ratio at most 2 [13], and that, no online deterministic algorithm can beat this 2 bound [15]. To the best of our knowledge, however, online coloring of unit co-interval graphs has not been studied before.

2 Naïve Algorithms

In this section, we begin our study of the unit clustering problem in one dimension by pointing out the deficiencies of some natural strategies.

Recall that the goal is to assign points to clusters so that each cluster has length at most 1, where the *length* of a cluster refers to the length of its smallest enclosing interval. (Note that we have switched to using lengths instead of radii in one dimension; all intervals are closed.) We say that a point *lies* in a cluster if inserting it to the cluster would not increase the length of the cluster. We say that a point *fits* in a cluster if inserting it to the cluster would not cause the length to exceed 1. The following are three simple online algorithms, all easily provable to have competitive ratio at most 2:

Algorithm 1 (Centered) *For each new point p, if it is covered by an existing interval, put p in the corresponding cluster, else open a new cluster for the unit interval centered at p.*

Algorithm 2 (Grid) *Build a uniform unit grid on the line (where cells are intervals of the form $[i, i+1)$). For each new point p, if the grid cell containing p is nonempty, put p in the corresponding cluster, else open a new cluster for the grid cell.*

Algorithm 3 (Greedy) *For each new point p, if p fits in some existing cluster, put p in such a cluster, else open a new cluster for p.*

The first two algorithms actually solve the stronger unit covering problem (Problem 2). No such algorithms can break the 2 bound, as we can easily prove:

Theorem 1. *There is a lower bound of 2 on the competitive ratio of any randomized (and deterministic) algorithm for the online unit covering problem in one dimension.*

Proof. To show the lower bound for randomized algorithms, we use Yao's technique and provide a probability distribution on the input sequences such that the resulting expected competitive ratio for any deterministic online algorithm is at least 2. The adversary provides a sequence of 3 points at position 1, x, and $1 + x$, where x is uniformly distributed in $[0, 1]$. The probability that a deterministic algorithm produces the optimal solution (of size 1 instead of 2 or more) is 0. Thus, the expected value of the competitive ratio is at least 2. □

The 2 bound on the competitive ratio is also tight for Algorithm 3: just consider the sequence $\left\langle \frac{1}{2}, \frac{3}{2}, \ldots, 2k - \frac{1}{2} \right\rangle$ followed by $\langle 0, 2, \ldots, 2k \rangle$ (where the greedy algorithm uses $2k + 1$ clusters and the optimal solution needs only $k + 1$ clusters). No random combination of Algorithms 1–3 can lead to a better competitive ratio, as we can easily see by the same bad example. New ideas are needed to beat 2.

3 The New Algorithm

In this section, we present a new randomized algorithm for the online unit clustering problem. While the competitive ratio of this algorithm is not necessarily less than 2, the algorithm is carefully designed so that when combined with Algorithm 2 we get a competitive ratio strictly less than 2.

Our algorithm builds upon the simple grid strategy (Algorithm 2). To guard against a bad example like $\langle \frac{1}{2}, \frac{3}{2}, \ldots \rangle$, the idea is to allow two points in different grid cells to be put in a common cluster "occasionally" (as controlled by randomization). Doing so might actually hurt, not help, in many cases, but fortunately we can still show that there is a net benefit (in expectation), at least in the most critical case.

To implement this idea, we form *windows* each consisting of two grid cells and permit clusters crossing the two cells within a window but try to "discourage" clusters crossing two windows. The details of the algorithm are delicate and are described below. Note that only one random bit is used at the beginning.

Algorithm 4 (RandWindow) *Group each two consecutive grid cells into a* window *of the form* $[2i, 2i+2)$. *With probability* $1/2$, *shift all windows one unit to the right. For each new point* p, *find the window* w *and the grid cell* c *containing* p, *and do the following:*

1: **if** w is empty **then** open a new cluster for p
2: **else if** p lies in a cluster **then** put p in that cluster
3: **else if** p fits in a cluster entirely inside c **then** put p in that cluster
4: **else if** p fits in a cluster intersecting w **then** put p in that cluster
5: **else if** p fits in a cluster entirely inside a neighboring window w' and
6: w' intersects > 1 clusters **then** put p in that cluster
7: **else** open a new cluster for p

To summarize: the algorithm is greedy-like and opens a new cluster only if no existing cluster fits. The main exception is when the new point is the first point in a window (line 1); another exception arises from the (seemly mysterious) condition in line 6. When more than one cluster fits, the preference is towards clusters entirely inside a grid cell, and against clusters from neighboring windows. These exceptional cases and preference rules are vital to the analysis.

4 Analysis

For a grid cell (or a group of cells) x, the *cost* of x denoted by $\mu(x)$ is defined to be the number of clusters fully contained in x plus half the number of clusters crossing the boundaries of x, in the solution produced by our algorithm. Observe that μ is additive, i.e., for two adjacent groups of cells x and y, $\mu(x \cup y) = \mu(x) + \mu(y)$. This definition of cost will be useful for accounting purposes.

To prepare for the analysis, we first make several observations concerning the behavior of the RandWindow algorithm. In the following, we refer to a cluster as a *crossing cluster* if it intersects two adjacent grid cells, or as a *whole cluster* if it is contained completely in a grid cell.

Observation 1

(i) *The enclosing intervals of the clusters are disjoint.*
(ii) *No grid cell contains two whole clusters.*
(iii) *If a grid cell c intersects a crossing cluster u_1 and a whole cluster u_2, then u_2 must be opened after u_1 has been opened, and after u_1 has become a crossing cluster.*

Proof. (i) holds because of line 2. (ii) holds because line 3 precedes line 7.

For (iii), let p_1 be the first point of u_1 in c and p_1' be the first point of u_1 in a cell adjacent to c. Let p_2 be the first point of u_2. Among these three points, p_1 cannot be the last to arrive: otherwise, p_1 would be assigned to the whole cluster u_2 instead of u_1, because line 3 precedes lines 4–7. Furthermore, p_1' cannot be the last to arrive: otherwise, p_1 would be assigned to u_2 instead, again because line 3 precedes lines 4–7. So, p_2 must be the last to arrive. □

For example, according to Observation 1(ii), every grid cell c must have $\mu(c) \leq 1 + \frac{1}{2} + \frac{1}{2} = 2$.

Let σ be the input sequence and $\texttt{opt}(\sigma)$ be an optimal covering of σ by unit intervals, with the property that the intervals are disjoint. (This property is satisfied by some optimal solution, simply by repeatedly shifting the intervals to the right.) We partition the grid cells into blocks, where each *block* is a maximal set of consecutive grid cells interconnected by the intervals from $\texttt{opt}(\sigma)$ (see Fig. 1). Our approach is to analyze the cost of the solution produced by our algorithm within each block separately.

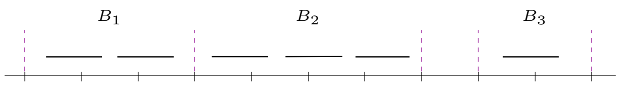

Fig. 1. Three blocks of sizes 2, 3, and 1.

A block of size $k \geq 2$ contains exactly $k-1$ intervals from $\texttt{opt}(\sigma)$. Define $\rho(k)$ to be the competitive ratio of the RandWindow algorithm within a block of size k, i.e., $\rho(k)$ upper-bounds the expected value of $\mu(B)/(k-1)$ over all blocks B of size k. The required case analysis is delicate and is described in detail below. The main case to watch out for is $k = 2$: any bound for $\rho(2)$ strictly smaller than 2 will lead to a competitive ratio strictly smaller than 2 for the final algorithm (as we will see in Section 5), although bounds for $\rho(3), \rho(4), \ldots$ will affect the final constant.

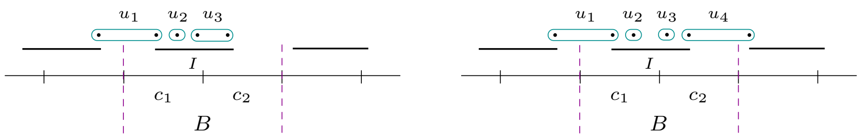

Fig. 2. Impossibility of Subcase 1.1 (left) and Subsubcase 1.3.2 (right).

Theorem 2. *$\rho(2) = 7/4$, $\rho(3) = 9/4$, $\rho(4) \le 7/3$, and $\rho(k) \le 2k/(k-1)$ for all $k \ge 5$.*

Proof. We first analyze $\rho(2)$. Consider a block B of size 2, consisting of cells c_1 and c_2 from left to right. Let I be the single unit interval in B in $\texttt{opt}(\sigma)$. There are two possibilities:

- Lucky Case: B falls completely in one window w. After a cluster u has been opened for the new point (by line 1), all subsequent points in I are put in the same cluster u (by lines 3 and 4). Note that the condition put in line 6 prevents points from the neighboring windows to join u and make crossing clusters. So, u is the only cluster in B, and hence, $\mu(B) = 1$.
- Unlucky Case: B is split between two neighboring windows. We first rule out some subcases:
  - Subcase 1.1: $\mu(c_1) = 2$. Here, c_1 intersects three clusters $\langle u_1, u_2, u_3 \rangle$ (from left to right), where u_1 and u_3 are crossing clusters and u_2 is a whole cluster (see Fig. 2, left). By Observation 1(iii), u_2 is opened after u_3 has become a crossing cluster, but then the points of u_2 would be assigned to u_3 instead (because line 4 precedes line 7 and $u_2 \cup u_3 \subset I$ has length at most 1): a contradiction.
  - Subcase 1.2: $\mu(c_2) = 2$. Similarly impossible.
  - Subcase 1.3: $\mu(c_1) = \mu(c_2) = 3/2$. We have only two scenarios:
    - ∗ Subsubcase 1.3.1: B intersects three clusters $\langle u_1, u_2, u_3 \rangle$, where u_2 is a crossing cluster, and u_1 and u_3 are whole clusters. By Observation 1(iii), u_1 is opened after u_2 has become a crossing cluster, but then the points of u_1 would be assigned to u_2 instead (because of line 4 and $u_1 \cup u_2 \subset I$): a contradiction.
    - ∗ Subsubcase 1.3.2: B intersects four clusters $\langle u_1, u_2, u_3, u_4 \rangle$, where u_1 and u_4 are crossing clusters and u_2 and u_3 are whole clusters (see Fig. 2, right). W.l.o.g., say u_2 is opened after u_3. By Observation 1(iii), u_2 is the last to be opened after u_1, u_3, u_4, but then u_2 would not be opened as points in u_2 may be assigned to u_3 (because lines 5–6 precedes line 7, $u_2 \cup u_3 \subset I$, and c_2 intersects more than one cluster): a contradiction.

  In all remaining subcases, $\mu(B) = \mu(c_1) + \mu(c_2) \le \frac{3}{2} + 1 = \frac{5}{2}$.

Since the lucky case occurs with probability exactly 1/2, we conclude that $\rho(2) \le \frac{1}{2}(1) + \frac{1}{2}(\frac{5}{2}) = \frac{7}{4}$. (This bound is tight.)

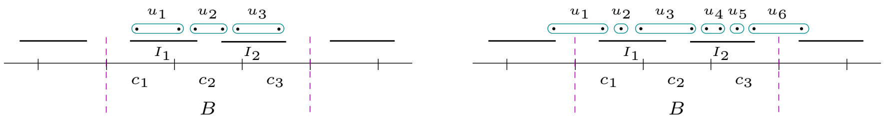

Fig. 3. Impossibility of Cases 2.1 (left) and 2.2 (right).

Next, we analyze $\rho(3)$. Consider a block B of size 3, consisting of cells c_1, c_2, c_3 from left to right. (It will not matter below whether c_1 and c_2 fall in the same window, or c_2 and c_3 instead.) Let I_1, I_2 be the two unit intervals in B in $\texttt{opt}(\sigma)$ from left to right.

- CASE 2.1: $\mu(c_2) = 2$. Here, c_2 intersects three clusters $\langle u_1, u_2, u_3 \rangle$ (from left to right), where u_1 and u_3 are crossing clusters and u_2 is a whole cluster (see Fig. 3, left). By Observation 1(iii), u_2 is opened after u_1 and u_3 have become crossing clusters, but then the points of u_2 would be assigned to u_1 or u_3 instead (because of line 4 and $u_1 \cup u_2 \cup u_3 \subset I_1 \cup I_2$): a contradiction.
- CASE 2.2: $\mu(c_1) = \mu(c_3) = 2$. Here, c_1 intersects three clusters $\langle u_1, u_2, u_3 \rangle$ and c_3 intersects three clusters $\langle u_4, u_5, u_6 \rangle$ (from left to right), where u_1, u_3, u_4, u_6 are crossing clusters and u_2, u_5 are whole clusters (see Fig. 3, right). Then u_3 cannot be entirely contained in I_1: otherwise, by Observation 1(iii), u_2 is opened after u_1 and u_3 have become crossing clusters, but then the points of u_2 would be assigned to u_3 instead. Similarly, u_4 cannot be entirely contained in I_2. However, this implies that the enclosing intervals of u_3 and u_4 overlap: a contradiction.
- CASE 2.3: $\mu(c_1) = 2$ and $\mu(c_2) = \mu(c_3) = 3/2$. Here, B intersects six clusters $\langle u_1, \ldots, u_6 \rangle$ (from left to right), where u_1, u_3, u_6 are crossing clusters and u_2, u_4, u_5 are whole clusters. As in Case 2.2, u_3 cannot be entirely contained in I_1. This implies that $u_4 \cup u_5 \subset I_2$. We now proceed as in Subcase 1.3.2. Say u_4 is opened after u_5 (the other scenario is symmetric). By Observation 1(iii), u_4 is the last to be opened after u_3, u_5, u_6, but then u_4 would not be opened as points in u_4 may be assigned to u_5: a contradiction.
- CASE 2.4: $\mu(c_1) = \mu(c_2) = 3/2$ and $\mu(c_3) = 2$. Similarly impossible.

In all remaining subcases, $\mu(B) = \mu(c_1) + \mu(c_2) + \mu(c_3)$ is at most $2 + \frac{3}{2} + 1 = \frac{9}{2}$ or $\frac{3}{2} + \frac{3}{2} + \frac{3}{2} = \frac{9}{2}$. We conclude that $\rho(3) \leq 9/4$. (This bound is tight.)

Now, we analyze $\rho(4)$. Consider a block B of size 4, consisting of cells $c_1, \ldots, c_4$ from left to right. Let I_1, I_2, I_3 be the three unit intervals in B in $\texttt{opt}(\sigma)$ from left to right.

- CASE 3.1: $\mu(c_1) = \mu(c_3) = 2$. Here, c_1 intersects three clusters $\langle u_1, u_2, u_3 \rangle$ and c_3 intersects three clusters $\langle u_4, u_5, u_6 \rangle$ (from left to right), where u_1, u_3, u_4, u_6 are crossing clusters and u_2, u_5 are whole clusters. As in Case 2.2, u_3 cannot be entirely contained in I_1. Thus, $u_4 \cup u_5 \cup u_6 \subset I_2 \cup I_3$. We now proceed as in Case 2.1. By Observation 1(iii), u_5 is opened after u_4 and u_6

have become crossing clusters, but then the points of u_5 would be assigned to u_4 or u_6 instead: a contradiction.

– CASE 3.2: $\mu(c_2) = \mu(c_4) = 2$. Similarly impossible.

In all remaining subcases, $\mu(B) = (\mu(c_1) + \mu(c_3)) + (\mu(c_2) + \mu(c_4)) \leq (2 + \frac{3}{2}) + (2 + \frac{3}{2}) \leq 7$. We conclude that $\rho(4) \leq 7/3$.

For $k \geq 5$, we use a rather loose upper bound. Consider a block B of size k. As each cell c has $\mu(c) \leq 2$, we have $\mu(B) \leq 2k$, and hence $\rho(k) \leq 2k/(k-1)$. □

5 The Combined Algorithm

We can now combine the RANDWINDOW algorithm (Algorithm 4) with the GRID algorithm (Algorithm 2) to obtain a randomized online algorithm with competitive ratio strictly less than 2. Note that only two random bits in total are used at the beginning.

Algorithm 5 (COMBO) *With probability* $1/2$, *run* RANDWINDOW, *else run* GRID.

Theorem 3. COMBO *is* $15/8$*-competitive (against oblivious adversaries).*

Proof. The GRID algorithm uses exactly k clusters on a block of size k. Therefore, the competitive ratio of this algorithm within a block of size k is $k/(k-1)$.

The following table shows the competitive ratio of the RANDWINDOW, GRID, and COMBO algorithms, for all possible block sizes.

Block Size	2	3	4	$k \geq 5$
GRID	2	3/2	4/3	$k/(k-1)$
RANDWINDOW	7/4	9/4	$\leq 7/3$	$\leq 2k/(k-1)$
COMBO	15/8	15/8	$\leq 11/6$	$\leq 3/2 \cdot k/(k-1)$

Table 1. The competitive ratio of the algorithms within a block.

As we can see, the competitive ratio of COMBO within a block is always at most 15/8. By summing over all blocks and exploiting the additivity of our cost function μ, we see that expected total cost of the solution produced by COMBO is at most 15/8 times the size of $\texttt{opt}(\sigma)$ for every input sequence σ. □

We complement the above result with a quick lower bound argument:

Theorem 4. *There is a lower bound of* $4/3$ *on the competitive ratio of any randomized algorithm for the online unit clustering problem in one dimension (against oblivious adversaries).*

Proof. We use Yao's technique. Consider two point sequences $P_1 = \langle 1, 2, \frac{1}{2}, \frac{5}{2} \rangle$ and $P_2 = \langle 1, 2, \frac{3}{2}, \frac{3}{2} \rangle$. With probability 2/3 the adversary provides P_1, and with probability 1/3 it provides P_2. Consider a deterministic algorithm $\mathcal{A}$. Regardless of which point sequence is selected by the adversary, the first two points provided to $\mathcal{A}$ are the same. If $\mathcal{A}$ clusters the first two points into one cluster, then it uses 3 clusters for P_1 and 1 cluster for P_2, giving the expected competitive ratio of $\frac{2}{3}(\frac{3}{2}) + \frac{1}{3}(1) = \frac{4}{3}$. If $\mathcal{A}$ clusters the first two points into two distinct clusters, then no more clusters are needed to cover the other two points of P_1 and P_2. Thus, the expected competitive ratio of $\mathcal{A}$ in this case is $\frac{2}{3} \cdot (1) + \frac{1}{3} \cdot (2) = \frac{4}{3}$ as well. □

6 Beyond One Dimension

In the two-dimensional L_∞-metric case, we want to partition the given point set into subsets, each of L_∞-diameter at most 1 (i.e., each enclosable by a unit square), so as to minimize the number of subsets used. (See Fig. 4.)

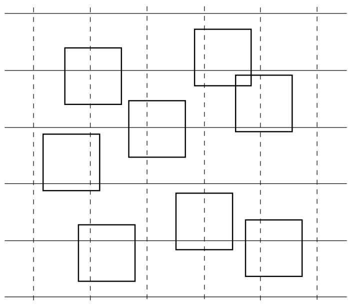

Fig. 4. Unit clustering in the L_∞ plane.

All the naïve algorithms mentioned in Section 2, when extended to two dimensions, provide 4-competitive solutions to the optimal solution. Theorem 1 can be generalized to a deterministic lower bound of 4 on the competitive ratio for the unit covering problem. We show how to extend Theorem 3 to obtain a competitive ratio strictly less than 4 for unit clustering.

Theorem 5. *There is a 15/4-competitive algorithm for the online unit clustering problem in the L_∞ plane.*

Proof. Our online algorithm is simple: just use Combo to find a unit clustering C_i for the points inside each horizontal strip $i \leq y < i+1$. (Computing each C_i is indeed a one-dimensional problem.)

Let σ be the input sequence. We denote by σ_i the set of points from σ that lie in the strip $i \leq y < i+1$. Let Z_i be an optimal unit covering for σ_i. Let O be an optimal unit covering for σ, and O_i be the set of unit squares in O that intersect the grid line $y = i$. Since all squares in O_i lie in the strip $i-1 \leq y < i+1$, we have

$|Z_i| \le |O_{i-1}| + |O_i|$. Therefore $\sum_i |Z_i| \le 2|O|$, so $\sum_i |C_i| \le \frac{15}{8} \sum_i |Z_i| \le \frac{15}{4}|O|$. □

The above theorem can easily be extended to dimension $d > 2$, with ratio $2^d \cdot 15/16$.

7 Closing Remarks

We have shown that determining the best competitive ratio for the online unit clustering problem is nontrivial even in the simplest one-dimensional case. The obvious open problem is to close the gap between the 15/8 upper bound and 4/3 lower bound. An intriguing possibility that we haven't ruled out is whether a nontrivial result can be obtained without randomization at all. There is an obvious 3/2 deterministic lower bound, but we do not see any simple argument that achieves a lower bound of 2.

We wonder if ideas that are more "geometric" may lead to still better results than Theorem 5. Our work certainly raises countless questions concerning the best competitive ratio in higher-dimensional cases, for other metrics besides L_∞, and for other geometric measures of cluster sizes besides radius or diameter.

References

1. U. Adamy and T. Erlebach. Online coloring of intervals with bandwidth. In *Proc. 1st Workshop Approx. Online Algorithms*, volume 2909 of *Lecture Notes Comput. Sci.*, pages 1–12, 2003.
2. P. K. Agarwal, M. van Kreveld, and S. Suri. Label placement by maximum independent set in rectangles. *Comput. Geom. Theory Appl.*, 11:209–218, 1998.
3. T. M. Chan. Polynomial-time approximation schemes for packing and piercing fat objects. *J. Algorithms*, 46:178–189, 2003.
4. M. Charikar, C. Chekuri, T. Feder, and R. Motwani. Incremental clustering and dynamic information retrieval. *SIAM J. Comput.*, 33(6):1417–1440, 2004.
5. M. Charikar, L. O'Callaghan, and R. Panigrahy. Better streaming algorithms for clustering problems. In *Proc. 35th ACM Sympos. Theory Comput.*, pages 30–39, 2003.
6. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms.* MIT Press, Cambridge, MA, 2nd edition, 2001.
7. L. Epstein and M. Levy. Online interval coloring and variants. In *Proc. 32nd International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 3580 of *Lecture Notes Comput. Sci.*, pages 602–613, 2005.
8. T. Erlebach, K. Jansen, and E. Seidel. Polynomial-time approximation schemes for geometric intersection graphs. *SIAM J. Comput.*, 34:1302–1323, 2005.
9. T. Feder and D. H. Greene. Optimal algorithms for approximate clustering. In *Proc. 20th ACM Sympos. Theory Comput.*, pages 434–444, 1988.
10. R. J. Fowler, M. S. Paterson, and S. L. Tanimoto. Optimal packing and covering in the plane are NP-complete. *Inform. Process. Lett.*, 12(3):133–137, 1981.
11. T. Gonzalez. Covering a set of points in multidimensional space. *Inf. Process. Lett.*, 40:181–188, 1991.

12. S. Guha, N. Mishra, R. Motwani, and L. O'Callaghan. Clustering data streams. In *Proc. 41st IEEE Sympos. Found. Comput. Sci.*, pages 359–366, 2000.
13. A. Gyárfás and J. Lehel. On-line and First-Fit colorings of graphs. *J. Graph Theory*, 12:217–227, 1988.
14. D. S. Hochbaum and W. Maas. Approximation schemes for covering and packing problems in image processing and VLSI. *J. ACM*, 32:130–136, 1985.
15. H. A. Kierstead and J. Qin. Coloring interval graphs with First-Fit. *SIAM J. Discrete Math.*, 8:47–57, 1995.
16. H. A. Kierstead and W. A. Trotter. An extremal problem in recursive combinatorics. *Congressus Numerantium*, 33:143–153, 1981.
17. R. J. Lipton and A. Tomkins. Online interval scheduling. In *Proc. 5th Sympos. Discrete Algorithms*, pages 302–311, 1994.
18. M. V. Marathe, H. Breu, H. B. Hunt III, S. S. Ravi, and D. J. Rosenkrantz. Simple heuristics for unit disk graphs. *Networks*, 25:59–68, 1995.
19. N. Megiddo and K. J. Supowit. On the complexity of some common geometric location problems. *SIAM J. Comput.*, 13(1):182–196, 1984.