Theoretical Computer Science 555 (2014) 43–54

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

A competitive strategy for distance-aware online shape allocation

Sándor P. Fekete [*], Jan-Marc Reinhardt, Nils Schweer

Department of Computer Science, TU Braunschweig, Germany

ARTICLE INFO

Available online 7 March 2014

Keywords:
Clustering
Average distance
Online problems
Optimal shape of a city
Space-filling curves
Competitive analysis

ABSTRACT

We consider the following online allocation problem: Given a unit square S, and a sequence of numbers $n_i \in [0,1]$ with $\sum_{j=0}^{i} n_j \leqslant 1$; at each step i, select a region C_i of previously unassigned area n_i in S. The objective is to make these regions compact in a distance-aware sense: minimize the maximum (normalized) average Manhattan distance between points from the same set C_i. Related location problems have received a considerable amount of attention; in particular, the problem of determining the "optimal shape of a city", i.e., allocating a *single* n_i has been studied. We present an online strategy, based on an analysis of space-filling curves; for continuous shapes, we prove a factor of 1.8092, and 1.7848 for discrete point sets.

1. Introduction

Many optimization problems deal with allocating point sets to a given environment. Frequently, the problem is to find compact allocations, placing points from the same set closely together. One well-established measure is the average L_1 distance between points. A practical example occurs in the context of grid computing, where one needs to assign a sequence of jobs i, each requiring an (appropriately normalized) number n_i of processors, to a subset C_i of nodes of a large square grid, such that the average communication delay between nodes of the same job is minimized; this delay corresponds to the number of grid hops [11], so the task amounts to finding subsets with a small average L_1, i.e., *Manhattan* distance. Karp et al. [8] studied the same problem in the context of memory allocation.

Even in an offline setting without occupied nodes, finding an optimal allocation for one set of size n_i is not an easy task; as shown in Fig. 1, the results are typically "round" shapes. If a whole sequence of sets have to be allocated, packing such shapes onto the grid will produce gaps, causing later sets to become disconnected, and thus leads to extremely bad average distances. Even restricting the shapes to be rectangular is not a remedy, as the resulting problem of deciding whether a set of squares (which are minimal with respect to L_1 average distance among all rectangles) can be packed into a given square container is NP-hard [10]; moreover, disconnected allocations may still occur.

In this paper, we give a first algorithmic analysis for the *online* problem. Using an allocation scheme based on a space-filling curve, we establish competitive factors of 1.8092 and 1.7848 for minimizing the maximum average Manhattan distance within an allocated set.

[*] Corresponding author.
E-mail addresses: s.fekete@tu-bs.de (S.P. Fekete), j-m.reinhardt@tu-bs.de (J.-M. Reinhardt), n.schweer@tu-bs.de (N. Schweer).

http://dx.doi.org/10.1016/j.tcs.2014.02.050
0304-3975/© 2014 Elsevier B.V. All rights reserved.

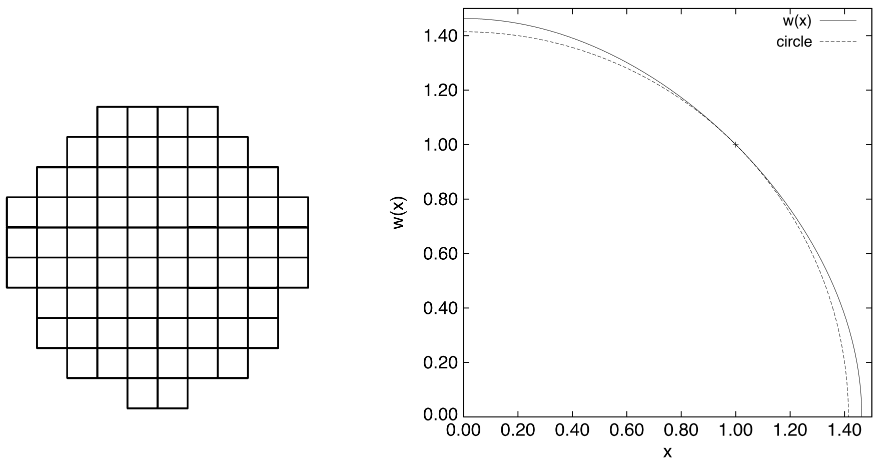

Fig. 1. Finding optimal individual shapes. (Left) An optimal shape composed of $n = 72$ grid cells, according to [5]. (Right) The optimal limit curve $w(x)$, according to [2].

1.1. Related work

Compact location problems have received a considerable amount of attention. Krumke et al. [9] have considered the *offline* problem of choosing a set of n vertices in a weighted graph, such that the average distance is minimized. They showed that the problem is NP-hard (even to approximate); for the scenario in which distances satisfy the triangle inequality, they gave algorithms that achieve asymptotic approximation factors of 2. For points in two-dimensional space with distances measured according to the Manhattan metric, Bender et al. [2] gave a simple 1.75-approximation algorithm, and a polynomial-time approximation scheme for any fixed dimension.

The problem of finding the "optimal shape of a city", i.e., a shape of given area that minimizes the average Manhattan distance, was first considered by Karp, McKellar, and Wong [8]; independently, Bender, Bender, Demaine, and Fekete [1] showed that this shape can be characterized by a differential equation for which no closed form is known. For the case of a finite set of n points that needs to be allocated to a grid, Demaine et al. [5] showed that there is an $O(n^{7.5})$-time dynamic-programming algorithm, which allowed them to compute all optimal shapes up to $n = 80$. Note that all these results are strictly offline, even though the original motivation (register or processor allocation) is online.

Space-filling curves for processor allocation with our objective function have been used before, see Leung et al. [11]; however, no algorithmic results and no competitive factor was proven. Wattenberg [16] proposed an allocation scheme similar to ours, for purposes of minimizing the maximum *Euclidean diameter* of an allocated shape. Like other authors before (in particular, Niedermeier et al. [12] and Gotsman and Lindenbaum [7]), he considered a measure called *c-locality*: for a sequence $1, \dots, i, \dots, j, \dots$ of points on a line, a space-filling mapping $h(.)$ will guarantee $L_2(h(i), h(j)) < c\sqrt{|j-i|}$, for a constant c that is $\sqrt{6} \approx 2.449$ for the Hilbert curve, and 2 for the so-called H-curve. One can use c-locality for establishing a constant competitive factor for our problems; however, given that the focus is on bounding the worst-case distance ratio for an embedding instead of the average distance, it should come as no surprise that the resulting values are significantly worse than the ones we achieve. On a different note, de Berg, Speckmann, and van der Weele [4] consider treemaps with bounded aspect ratio. Other related work includes Dai and Su [3].

1.2. Our results

We give a first competitive analysis for the online shape allocation problem within a given bounding box, with the objective of minimizing the maximum average Manhattan distance. In particular, we give the following results.

- We show that for the case of continuous shapes (in which numbers n_i correspond to area), a strategy based on a space-filling Hilbert curve achieves a competitive ratio of 1.8092.
- For the case of discrete point sets (in which numbers n_i indicate the number of points that have to be chosen from an appropriate $N \times N$ orthogonal grid), we prove a competitive factor of 1.7848.
- We sketch how these factors may be further improved, but point out that a Hilbert-based strategy is no better than a competitive factor of 1.3504, even with an improved analysis.
- We establish a lower bound of 1.144866 for *any* online strategy in the case of discrete point sets, and argue the existence of a lower bound for the continuous case.

The rest of this paper is organized as follows. In Section 2, we give some basic definitions and fundamental facts. In Section 3, we provide a brief description of an allocation scheme based on a space-filling curve. Section 4 provides more technical details for the allocation. Section 5 gives a mathematical study for the case of continuous allocations, proving that the analysis can be reduced to a limited number of shapes, and establishes a competitive factor of 1.8092. Section 6 sketches a similar analysis for the case of discrete allocations; as a result, we prove a competitive factor of 1.7848. Section 7 discusses lower bounds for online strategies. Final conclusions are presented in Section 8.

2. Preliminaries

We examine the problem of selecting shapes from a square, such that the maximum average L_1-distance of the shapes is minimized. We first formulate the problem more precisely. This covers both the continuous and the discrete case; the former arises as the limiting case of the latter, while the latter needs to be considered for allocations within a grid of limited size.

Definition 1. A *city* is a (continuous) shape in the plane with fixed area. For a city C of area n, we call

$$c(C) = \frac{1}{2} \iiiint_{(x,y),(u,v)\in C} \big(|x-u| + |y-v|\big)\,\mathrm{d}v\,\mathrm{d}u\,\mathrm{d}y\,\mathrm{d}x \tag{1}$$

the *total Manhattan distance between all pairs of points in C* and

$$\phi(C) = \frac{2\,c(C)}{n^{5/2}} \tag{2}$$

the *ϕ-value* or *average distance* of C. An *n-town* T is a subset of n points in the integer grid. Its *normalized average Manhattan distance* is

$$\phi(T) = \frac{2c(T)}{n^{5/2}} = \frac{\sum_{s\in T}\sum_{t\in T}\|s-t\|_1}{n^{5/2}} \tag{3}$$

The normalization with $n^{2.5}$ yields a dimensionless measure that remains unchanged under scaling, and makes the continuous and the discrete case comparable; see [1].

Problem 2. In the continuous setting, we are given a sequence $n_1, n_2, \ldots, n_k \in \mathbb{R}^+$ with $\sum_{i=1}^k n_i \leqslant 1$. Cities $C_1, C_2, \ldots, C_k$ of size $n_1, n_2, \ldots, n_k$ are to be chosen from the unit square, such that $\max_{1\leqslant i\leqslant k} \phi(C_i)$ is minimized.

In the discrete setting, we are given a sequence $n_1, n_2, \ldots, n_k \in \mathbb{N}^+$ with $\sum_{i=1}^k n_i \leqslant N^2$. Towns $C_1, C_2, \ldots, C_k$ of size $n_1, n_2, \ldots, n_k$ are to be chosen from the $N \times N$ grid, such that $\max_{1\leqslant i\leqslant k} \phi(C_i)$ is minimized.

Although it has not been formally proven, the offline problem is conjectured to be NP-hard, see [14]; if we restrict city shapes to be rectangles, there is an immediate reduction from deciding whether a set of squares can be packed into a larger square [10]. (A special case arises from considering integers, which corresponds to choosing grid locations.) Our approximation works online, i.e., we choose the cities in a specified order, and no changes can be made to previously allocated cities; clearly, this implies approximation factors for the corresponding offline problems.

There are lower bounds for $\max_{1\leqslant i\leqslant k} \phi(C_i)$ that generally cannot be achieved by any algorithm. One important result is the following theorem.

Theorem 3. *Let C be any city. Then $\phi(C) \geqslant 0.650245$.*

A proof can be found in [1]. For $n_1 = 1$ any algorithm must select the whole unit square, thus $2/3$, the ϕ-value of a square, is a lower bound for the achievable ϕ-value. We will discuss better lower bounds in the conclusions.

3. An allocation strategy

While long and narrow shapes tend to have large ϕ-values, shapes that fill large parts of an enclosing rectangle with similar width and height usually have better average distances; however, one has to make sure that early choices with small average distance do not leave narrow pieces with high average distance, or even disconnected pieces, making the normalized ϕ-values potentially unbounded.

Our approach uses the recursive Hilbert family of curves in order to yield a provably constant competitive factor. That family is based on a recursive construction scheme and becomes space filling for infinite repetition of said scheme [13]. For a finite number r of repetitions, the curve traverses all points of the used grid. For $1 \leqslant r \leqslant 3$, the curve is shown in Fig. 2. Thus, the Hilbert curve provides an order for the cells of the grid, which is then used for allocation, as illustrated in Fig. 3. More formal details of the recursive definition of the Hilbert family (e.g. with text-rewriting rules, such as the ones in [15]) are cited and sketched in the following Section 4.

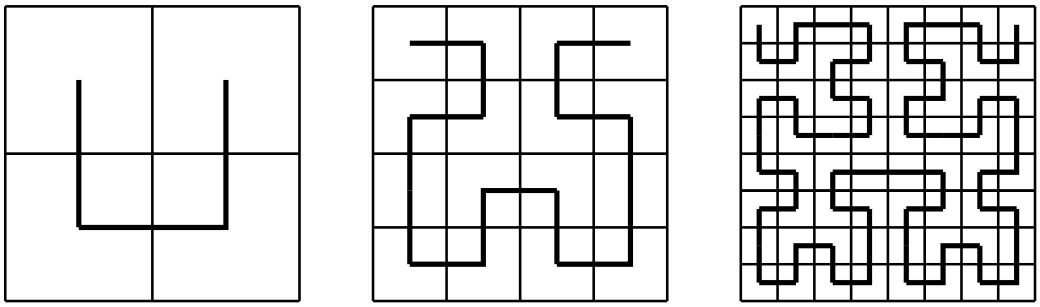
Fig. 2. Hilbert curve with $1 \leqslant r \leqslant 3$.

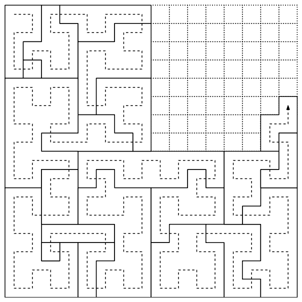
Fig. 3. A sample allocation according to our strategy.

More technically, the unit square is recursively subdivided into a grid consisting of $2^r \times 2^r$ grid cells, for an appropriate refinement level $r > 0$, as shown in Fig. 2. For the sake of concise presentation, we assume that every input n_i is an integral multiple of $c = 4^{-R}$, for an appropriately large $R > 0$. (We will mention in the Conclusions how this assumption can be removed, based on Lemma 11.) Similar to the recursive structure of quad-trees, the actual subdivision can be performed in a self-refining manner, whenever a grid cell is not completely filled. This means that during the course of the online allocation, we may use different refinement levels in different parts of the layout; however, this will not affect the overall analysis, as further refinement of the grid does not change the quality of existing shapes.

Definition 4. For a given refinement level r, an *r-pixel* P is a grid square of size $2^{-r} \times 2^{-r}$. For a given allocated shape C_i, a pixel is *full* if $P \subseteq C_i$; it is *fractional*, if $P \cap C_i \neq \emptyset$ and $P \not\subseteq C_i$.

Now the description of the algorithm is simple: for every input n_i, choose the next set of $n_i/2^R$ R-pixels traversed by the Hilbert curve as the city C_i, starting in the upper left corner of the grid. For an illustration, see Fig. 3.

4. Technical details of shape allocation

A technical different description of the Hilbert family can be based on the string representation given in [15]. There, the authors use the following recursion, based on the letters u, r, d, l denoting "up", "right", "down", and "left".

$$y_1 = \mathrm{urd}$$

$$y_n = h_4(y_{n-1})\mathrm{u}y_{n-1}\mathrm{r}y_{n-1}\mathrm{d}h_5(y_{n-1}) \quad \text{if } n > 1$$

where h_4 and h_5 are defined as

$$h_4(x) = \begin{cases} \mathrm{r} & \text{if } x = \mathrm{u}, \\ \mathrm{l} & \text{if } x = \mathrm{d}, \\ \mathrm{u} & \text{if } x = \mathrm{r}, \\ \mathrm{d} & \text{if } x = \mathrm{l}, \end{cases} \quad \text{and} \quad h_5(x) = \begin{cases} \mathrm{l} & \text{if } x = \mathrm{u}, \\ \mathrm{r} & \text{if } x = \mathrm{d}, \\ \mathrm{d} & \text{if } x = \mathrm{r}, \\ \mathrm{u} & \text{if } x = \mathrm{l}, \end{cases}$$

and $h_i(x_0x_1x_2\ldots) = h_i(x_0)h_i(x_1)h_i(x_2)\ldots$, $i \in \{4, 5\}$.

We combine those characters to four sequences of length three and build an L-system, a special kind of formal grammar with parallel rewriting. Let $A = \mathrm{dlu}$, $B = \mathrm{ldr}$, $C = \mathrm{rul}$, and $D = \mathrm{urd}$, as shown in Fig. 4. Using the production rules

$$A \to BAAC,$$

$$B \to ABBD,$$

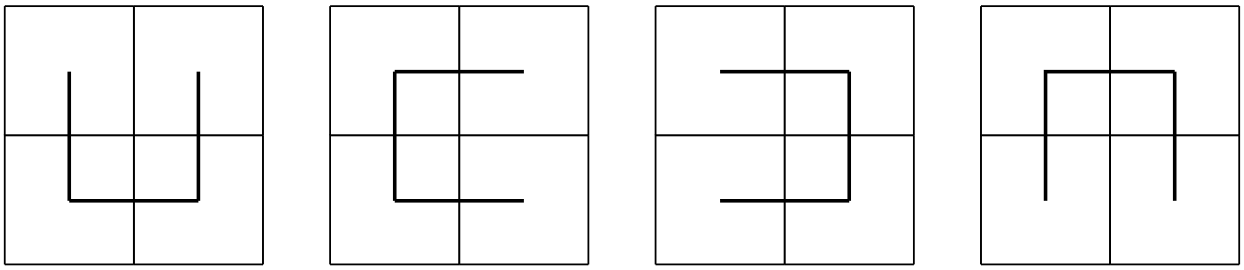
Fig. 4. A, B, C, and D.

$C \to DCCA$ and

$D \to CDDB$

and starting with D, you get the same string from [15] if you replace $X_0X_1X_2X_3$ by $X_0uX_1rX_2dX_3$, $X_0, X_1, X_2, X_3 \in \{A, B, C, D\}$, after each use of a production rule. Note that [15] describes the Hilbert curve as a curve from the lower left-hand corner to the lower right-hand corner of the unit square, while the Hilbert curve described in Section 3 starts in the upper left-hand corner and ends in the upper right-hand corner. However, this does not affect the shape of allocated cities.

A sequence of symbols produced by the L-system can be interpreted graphically as a sequence of subsquares $E^2_{r-1}[j][k]$, $j, k \in \{1, \dots, 2^{r-1}\}$, see Fig. 4. The figure also shows that A is D rotated by 180°. The same is true for B and C. We denote by $\overline{X}$ the symbol that is X rotated by 180°. As we are only interested in the shape of cities, we identify symmetric cities. Consequently, we make no distinction between X and $\overline{X}$ when we look at a single symbol. We write $X \equiv \overline{X}$. Similarly, we get equivalences for longer sequences of symbols: $XY \equiv YX$, because the order of the successive subsquares $E^2_{r-1}[j][k]$ does not change the shape. Furthermore, we have $X\overline{X} \equiv Y\overline{Y}$ for each X and Y, as it is always the simple shape followed by the rotated shape, and $XX \equiv \overline{X}\overline{X}$ for two occurrences of the same shape in succession.

Lemma 5. *After $r > 1$ uses of the production rules (where $r = 1$ means that we are still at the start symbol D), any subsequence of length 2^{r-2} that can be created with the L-system is contained as a symmetric copy in the resulting sequence.*

Proof. We prove the claim via induction over r:

$r = 2$: $CDDB$ contains C, B, and $D \equiv \overline{D} = A$.

$r = 3$: $DCCACDDBCDDBABBD$ contains
- $AB \equiv BA \equiv CD \equiv DC$
- $AC \equiv CA \equiv BD \equiv DB$
- $BC \equiv CB \equiv DA \equiv AD$
- $BB \equiv CC$
- $DD \equiv AA$.

Thus, for $r \in \{2, 3\}$, the resulting sequence contains all possible subsequences of length 2^{r-2}. In particular, it contains every subsequence of length 2^{r-2} that the L-system can generate.

$r \to r + 1$, $r \geqslant 3$: A sequence of length 2^{r-1} has been created from a sequence of length at most $2^{r-1}/4 + 1 = 2^{r-3} + 1$, because every symbol is replaced by exactly 4 symbols. If the claim holds for $r \geqslant 3$, then all sequences of length 2^{r-2} have been created, i.e., all sequences of length $2^{r-3} + 1$ have been created as well. Thus, in the next step the production rules are applied to all possible sequences of that length, yielding all sequences of length 2^{r-1} that can be created with the L-system. □

In the following, we denote by W_n the worst city consisting of n pixels that can be produced by our Hilbert strategy; because of the normalized nature of ϕ, this is independent of the size of the pixels, and only the shape matters.

Lemma 6. *A city W_n contains at most parts of $\lceil n/4 \rceil + 1$ subsquares $E^2_{r-1}[j][k]$, $j, k \in \{1, \dots, 2^{r-1}\}$, i.e., subsquares that consist of exactly 4 cells.*

Proof. A city W_n consists of exactly n cells. Assume that these cells belong to at least $\lceil n/4 \rceil + 2$ subsquares $E^2_{r-1}[j][k]$. Then at least three of those subsquares cannot belong to W_n as a whole but only in part, because n cells cannot completely fill more than $\lceil n/4 \rceil$ of the subsquares, and if two more are partially filled not even all those can be filled completely. Consider the sequence of subsquares of W_n in the order given by the Hilbert curve. One of the subsquares that is not the first or the last in the sequence cannot completely belong to W_n. This is a contradiction to the definition of the Hilbert curve, which recursively uses the same construction scheme for subsquares on every level of refinement. □

Lemma 7. *When the L-system has generated all sequences of length $\lceil n/4 \rceil + 1$, the resulting Hilbert curve traverses a city W_n.*

Proof. Each symbol of the L-system corresponds to a subsquare $E^2_{r-1}[j][k]$. Once all sequences of length $\lceil n/4 \rceil + 1$ symbols have been generated, all possible cities consisting of $\lceil n/4 \rceil + 1$ subsquares have been generated, too. With Lemma 6, the claim follows. □

Lemma 8. *For $r = \lceil \log_2(\lceil n/4 \rceil + 1) \rceil + 2$, the Hilbert curve traverses a city W_n.*

Proof. Using Lemmas 5 and 7, we know that the Hilbert curve traverses a city W_n, if the following holds:

$$2^{r-2} \geqslant \lceil n/4 \rceil + 1$$

This is true for

$$r = \left\lceil \log_2\left(\lceil n/4 \rceil + 1\right) \right\rceil + 2. \qquad \square$$

5. Analysis

For the analysis of our allocation scheme, we first establish a couple of technical lemmas. These will then allow it to "fill in" bigger pixels that are only partially filled by a city, which yields upper bounds for the total distance at a coarser refinement level. In a second step, this is used to derive global bounds by computing the worst-case bounds for shapes of at most refinement level 3; thus the computational results presented in Table 1 for shapes of limited size are not merely experiments, but yield a general upper bound on the competitive factor. (As discussed in the Conclusions, carrying out the computations on a lower or higher refinement level gives looser or tighter results.)

Lemma 9. *For any integer $m > 0$, let A be the set of the first m R-pixels formed by the Hilbert curve. Then at any refinement level $r \in \{0, \dots, R\}$, there is at most one fractional r-pixel, i.e., a pixel that contains p R-pixels of A with $0 < p < 4^{R-r}$.*

Proof. This claim follows as an immediate consequence of the recursive structure of the Hilbert curve: For any $r \in \{0, \dots, R\}$, the curve only leaves an $(R - r)$-pixel after filling it completely. □

This implies the following lemma.

Lemma 10. *Let C be a city generated by our strategy. Then at any refinement level $r \in \{0, \dots, R\}$, C contains at most two fractional r-pixels, i.e., two pixels that contain p R-pixels of C with $0 < p < 4^{(R-r)}$.*

Proof. Let A be the set of R-pixels allocated before C is placed. Consider an arbitrary $r \in \{0, \dots, R\}$. By Lemma 9, there is at most one fractional r-pixel P in A. By construction, the initial R-pixels of C will be placed in P, so P is fractional with respect to C. After placing C, we apply Lemma 9 to $A \cup C$ and conclude that there can be at most one further fractional r-pixel, proving the claim. □

Now we use this lemma for deriving upper bounds by filling in the (at most two) fractional r-pixels.

Lemma 11. *Let C be a city generated by our strategy with area at most $n \leqslant l 4^r 4^{-R}$ for $r \in \{0, 1, \dots, R\}$, $l \in \mathbb{N}$. Then we have $c(C) \leqslant c(W_{l+1})$, where W_{l+1} is a worst case among all cities produced by our allocation scheme that consists of $(l + 1)$ r-pixels.*

Proof. By Lemma 10, we know that only the first and the last r-pixel of C may be fractional. Therefore C cannot intersect more than $l + 1$ r-pixels. By replacing the two fractional pixels by full pixels, we get a city W that consists of $l + 1$ full r-pixels, and $c(C) \leqslant c(W)$. By definition, $c(W) \leqslant c(W_{l+1})$, and the claim holds. □

Therefore, we can give upper bounds for the worst case by considering the values of W_n at some moderate refinement level. The W_n can be found by enumeration; as described by the technical lemmas in preceding section, a speed-up can be achieved by making use of the recursive construction of the W_n. We determined the shapes and ϕ-values of the W_n for $n \leqslant 65$; by Lemma 11, this suffices to provide upper bounds for all cities with area up to $64 \cdot 2^{-r}$, i.e., these computational results give an estimate for the round-up error using refinement level 3. The full table of average distances for this refinement level can be seen in Table 1; the worst cases among the examined ones are W_{56} and W_{14}, which have the same shape, shown in Fig. 5.

Theorem 12. *Our strategy guarantees $\max_{1 \leqslant n \leqslant k} \phi(C_n) \leqslant 1.1764$.*

Table 1
Total and average distances for cities W_n allocated according to our strategy, as well as the optimal values $c_{city}(n)$ according to [5].

n	$c^*(W_n)$	$c_{city}(n)$	$\phi(W_n)$	$\Phi(W_n)$
1	0 1/3	0 1/3	0.6667	
2	2	2	0.7071	
3	6	5 2/3	0.7698	
4	13 1/3	10 2/3	0.8333	
5	24	19 2/3	0.8587	
6	38	30	0.8619	
7	50 2/3	44	0.7816	
8	74 2/3	61 1/3	0.8250	
9	102	81	0.8395	
10	134 2/3	106 1/3	0.8517	
11	162	135 2/3	0.8074	
12	210 2/3	165 1/3	0.8446	
13	262 2/3	203	0.8621	
14	322	244	0.8781	
15	371 2/3	290 2/3	0.8530	
16	434 2/3	338 2/3	0.8490	1.1764
17	512 2/3	396 1/3	0.8605	1.1497
18	602 1/3	457 1/3	0.8764	1.1174
19	685	522 1/3	0.8706	1.1058
20	768	591 2/3	0.8587	1.1098
21	870	663	0.8610	1.0903
22	992 2/3	749 1/3	0.8745	1.0713
23	1101 2/3	839 1/3	0.8685	1.0424
24	1216	933	0.8619	1.0150
25	1322 1/3	1032 1/3	0.8463	0.9777
26	1432	1134	0.8309	0.9701
27	1527 2/3	1249	0.8066	0.9785
28	1672	1365 1/3	0.8061	0.9864
29	1853 1/3	1492	0.8184	0.9773
30	2046	1622 2/3	0.8301	0.9710
31	2213	1759 2/3	0.8272	0.9726
32	2393 1/3	1898 2/3	0.8263	0.9791
33	2602	2057	0.8319	0.9735
34	2835 2/3	2216 2/3	0.8414	0.9691
35	3045	2384	0.8403	0.9712
36	3266	2554 2/3	0.8400	0.9772
37	3519 1/3	2727 2/3	0.8453	0.9726
38	3799 1/3	2921 2/3	0.8536	0.9682
39	4049 2/3	3117 2/3	0.8527	0.9570
40	4309 1/3	3322	0.8517	0.9463
41	4545	3530 1/3	0.8445	0.9307
42	4788	3749	0.8376	0.9214
43	5009	3976	0.8262	0.9306
44	5266 2/3	4205 1/3	0.8202	0.9393
45	5641 1/3	4456 2/3	0.8306	0.9393
46	6031 1/3	4712	0.8405	0.9395
47	6379 2/3	4970 1/3	0.8425	0.9439
48	6741 1/3	5234	0.8446	0.9505
49	7147 1/3	5507 1/3	0.8505	0.9512
50	7586 1/3	5788	0.8583	0.9516
51	7993	6076 1/3	0.8606	0.9559
52	8411 1/3	6368	0.8628	0.9620
53	8878	6691 2/3	0.8683	0.9619
54	9379 1/3	7017 1/3	0.8754	0.9617
55	9835	7352 1/3	0.8768	0.9569
56	10 304	7690	0.8781	0.9522
57	10 733	8033 2/3	0.8751	0.9445
58	11 173 1/3	8384 2/3	0.8723	0.9372
59	11 583 2/3	8749 1/3	0.8665	0.9268
60	12 005 1/3	9117 1/3	0.8610	0.9225
61	12 391	9506 1/3	0.8527	0.9232
62	12 862	9904 2/3	0.8499	0.9201
63	13 415	10 305 1/3	0.8517	0.9175
64	13 924 2/3	10 719 1/3	0.8499	
65	14 452 2/3	11 139 2/3	0.8486	

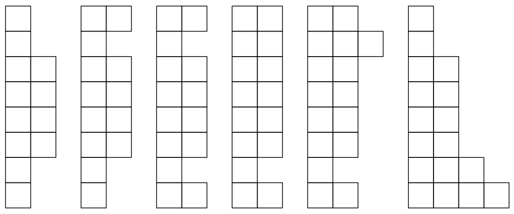

Fig. 5. Worst cases W_n for $12 \leqslant n \leqslant 17$.

Proof. Consider a city C of size n generated by our strategy. If n is sufficiently small, i.e., smaller than an r-pixel, $r \geqslant 0$, C consists of at most 4^r cells and its average distance can be bounded by the worst case for that particular number of cells. In the case that C has a larger, more complicated shape, an analysis of a finite number of shapes is still sufficient:

We know that $n > 4^j c$ and can assume that $n \leqslant 4^{j+1}c$ (or else we use the analysis on the less refined $E^2_{r-(j+1)}[p][q]$). Thus, there must be an l such that $l4^j c < n \leqslant (l+1)4^j c$ with $l = 1, \dots, 3$. Yet, we can get closer to n, as we know that $E^2_{r-j}[p][q]$ consists of 4^j cells. We get the inequality $l4^{j-k}c < n \leqslant (l+1)4^{j-k}c$, $k \leqslant j$, $l = 4^k, \dots, 4^{k+1} - 1$.

Hence, a city of arbitrary size n corresponds to at most $(l+1)$ subsquares of a certain size (depending on the precision of the analysis), i.e., a city of size at most $(l+1)4^{j-k}c$. Now we can use Lemma 10 to bound the average distance of the city, yielding

Table 2
Total distances for towns T_n allocated according to our strategy, and the corresponding optimal values c_{town} for n-towns.

n	$c(T_n)$	$c_{town}(n)$	$\rho(n)$	$\phi(T_n)$	$\Phi(T_n)$
1	0	0	–	0.0000	
2	1	1	1.0000	0.3536	
3	4	4	1.0000	0.5132	
4	10	8	1.2500	0.6250	
5	20	16	1.2500	0.7155	
6	33	25	1.3200	0.7485	
7	44	38	1.1579	0.6788	
8	66	54	1.2222	0.7292	
9	92	72	1.2778	0.7572	
10	123	96	1.2812	0.7779	
11	148	124	1.1935	0.7376	
12	194	152	1.2763	0.7778	
13	244	188	1.2979	0.8009	
14	301	227	1.3260	0.8209	
15	348	272	1.2794	0.7987	
16	410	318	1.2893	0.8008	1.1230
17	488	374	1.3048	0.8191	1.1001
18	575	433	1.3279	0.8366	1.0708
19	656	496	1.3226	0.8338	1.0626
20	736	563	1.3073	0.8229	1.0700
21	836	632	1.3228	0.8273	1.0530
22	957	716	1.3366	0.8431	1.0360
23	1064	804	1.3234	0.8388	1.0091
24	1176	895	1.3140	0.8335	0.9831
25	1280	992	1.2903	0.8192	0.9472
26	1387	1091	1.2713	0.8048	0.9388
27	1480	1204	1.2292	0.7814	0.9483
28	1618	1318	1.2276	0.7800	0.9570
29	1796	1442	1.2455	0.7931	0.9486
30	1985	1570	1.2643	0.8054	0.9437
31	2148	1704	1.2606	0.8029	0.9464
32	2326	1840	1.2641	0.8031	0.9540
33	2532	1996	1.2685	0.8095	0.9489
34	2763	2153	1.2833	0.8198	0.9453
35	2968	2318	1.2804	0.8191	0.9482
36	3186	2486	1.2816	0.8194	0.9550
37	3436	2656	1.2937	0.8252	0.9511
38	3713	2847	1.3042	0.8342	0.9473
39	3960	3040	1.3026	0.8338	0.9366
40	4216	3241	1.3008	0.8333	0.9263
41	4448	3446	1.2908	0.8265	0.9112
42	4687	3662	1.2799	0.8200	0.9017
43	4904	3886	1.2620	0.8089	0.9112
44	5154	4112	1.2534	0.8027	0.9203
45	5524	4360	1.2670	0.8133	0.9205
46	5909	4612	1.2812	0.8235	0.9212
47	6252	4868	1.2843	0.8257	0.9260
48	6610	5128	1.2890	0.8282	0.9331
49	7012	5398	1.2990	0.8344	0.9339
50	7447	5675	1.3122	0.8425	0.9347
51	7848	5960	1.3168	0.8450	0.9393
52	8262	6248	1.3223	0.8474	0.9458
53	8724	6568	1.3283	0.8532	0.9459
54	9221	6890	1.3383	0.8606	0.9460
55	9672	7222	1.3392	0.8623	0.9414
56	10 136	7556	1.3415	0.8638	0.9370
57	10 560	7896	1.3374	0.8610	0.9295
58	10 995	8243	1.3339	0.8583	0.9224
59	11 400	8604	1.3250	0.8527	0.9123
60	11 816	8968	1.3176	0.8475	0.9086
61	12 196	9354	1.3038	0.8393	0.9098
62	12 669	9749	1.2995	0.8371	0.9068
63	13 220	10 146	1.3030	0.8393	0.9051
64	13 724	10 556	1.3001	0.8376	
65	14 256	10 972	1.2993	0.8370	

$$\phi(C) \leqslant \frac{2c(W)}{(l4^{j-k}c)^{5/2}} = \frac{\phi(W_{l+2})((l+2)4^{j-k}c)^{5/2}}{(l4^{j-k}c)^{5/2}} \tag{4}$$

$$= \phi(W_{l+2})\left(1+\frac{2}{l}\right)^{5/2} =: \Phi(W_l). \tag{5}$$

The resulting bound is $\max(\{\phi(W_i)\colon\ 1 \leqslant i \leqslant 4^j\} \cup \{\Phi(W_l)\colon\ 4^k \leqslant l \leqslant 4^{k+1}-1\})$. Note that the number of shapes considered is at most 4^{k+1}.

We conducted the calculations for $k=2$ and list the results in Table 1. As it turns out, the maximum is attained for $\Phi(W_{16}) = 1.1764$. □

Corollary 13. *Our strategy achieves a competitive factor of 1.8092.*

Proof. According to Theorem 3, no algorithm can guarantee a better ϕ-value than 0.650245. Our strategy yields an upper bound of 1.1764. This results in a factor of $1.1764/0.650245 \approx 1.8092$. □

6. Discrete point sets

Our above analysis relies on continuous weight distributions, which imply the lower bound on ϕ-values stated in Theorem 3. This does not include the discrete scenario, in which the values n_i indicate a number of integer grid points that have to be chosen from an appropriate $N \times N$-grid. As discussed in the paper [5], considering discrete weight distributions may allow lower average distances; e.g., a single point yields a ϕ-value of 0. As a consequence, *towns* (subsets of the integer grid) have lower average distances than cities of the same total weight. However, we still get a competitive ratio for the case of online towns.

Theorem 14. *For n-towns, a Hilbert-curve strategy guarantees a competitive factor of at most 1.7848 for the ϕ-value.*

Proof. Lemma 10 still holds, so analogously to Theorem 12, we consider the values up to $n=64$, and show that the worst case is attained for $n=16$, which yields an upper bound of 1.123. See Table 2 for detailed numbers.

Table 3
ϕ-values of optimal n-towns, calculated using Table 1 from [5].

n	$\phi_{opt}(n)$	n	$\phi_{opt}(n)$	n	$\phi_{opt}(n)$	n	$\phi_{opt}(n)$
65	0.644217	69	0.643676	73	0.645275	77	0.645053
66	0.644281	70	0.644399	74	0.645136	78	0.645234
67	0.644240	71	0.645067	75	0.645072	79	0.645524
68	0.644104	72	0.645317	76	0.644715	80	0.645595

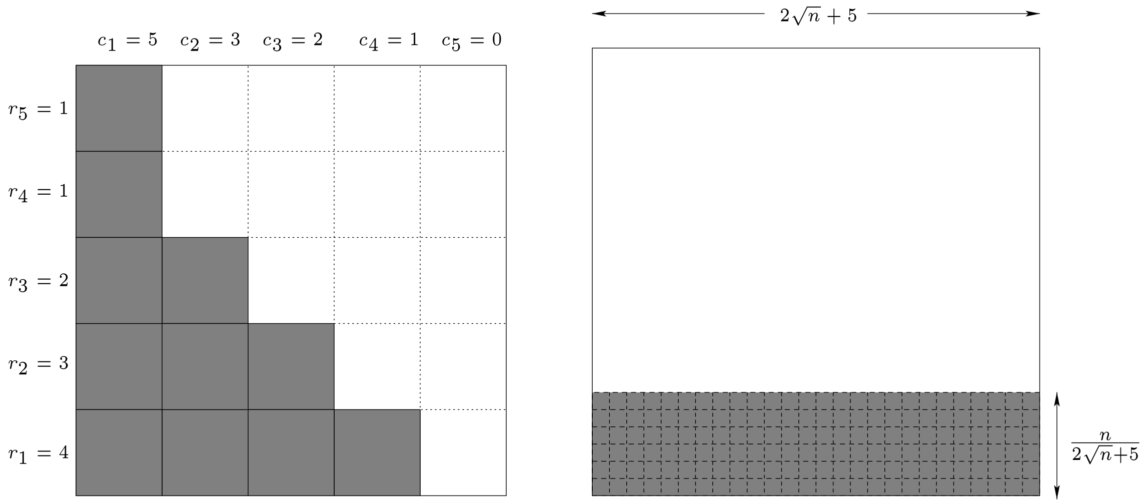

Fig. 6. Establishing a lower bound for ϕ: Defining $\Lambda(n)$; an arrangement that maximizes $\Lambda(n)$.

For a lower bound, the general value of 0.650245 for ϕ-values cannot be applied, as discrete point sets may have lower average distance. Instead, we verify that the ratio $\rho(n)$ of achieved ϕ to optimal ϕ, is less than 1.7848; this is the same as $c(T_n)/c_{town}(n)$ for $n \leqslant 64$, see Table 2. For $65 \leqslant n \leqslant 80$, the optimal values in [5] allow us to verify that $\phi \geqslant 0.6292$; see our Table 3.

Thus, we have to establish a lower bound for ϕ for $n \geqslant 81$. We make use of Eq. (5), p. 89 of [5]; see Fig. 6: for a given number n of grid points, the difference between the optimal total Manhattan distance $c_{city}(n)$ for a city consisting of n unit squares and the optimal total distance $c_{town}(n)$ for a town consisting of n grid points is equal to $\Lambda(n) := \frac{1}{6}(\sum_i c_i^2 + \sum_j r_j^2)$, where c_i is the number of grid points in column i, and r_j is the number of grid points in row j. Because $\frac{2c_{city}(n)}{n^{2.5}}$ is bounded from below by $\psi = 0.650245$, we get a lower bound of $\psi - \frac{2\Lambda(n)}{n^{2.5}} \leqslant \frac{2c_{town}(n)}{n^{2.5}}$ for the ϕ-value of an n-town.

This leaves the task of providing an upper bound for $2\Lambda(n)/n^{2.5}$. According to Lemma 5 of [5], the bounding box of an optimal n-town does not exceed $2\sqrt{n}+5$. Therefore, we have $c_i \leqslant 2\sqrt{n}+5$; as $\sum_i c_i = n$ and the function $\sum_i c_i^2$ is superlinear in the c_i, we conclude that $\sum_i c_i^2$ is maximized by subdividing n into $\frac{n}{2\sqrt{n}+5}$ columns of $2\sqrt{n}+5$ points each, so $\sum_i c_i^2 \leqslant n(2\sqrt{n}+5)$. Analogously, we have $\sum_j r_j^2 \leqslant n(2\sqrt{n}+5)$, so $2\Lambda(n)/n^{2.5} \leqslant \frac{2}{3}(\frac{2}{n} + \frac{5}{n^{1.5}})$. For $n \geqslant 81$, this implies $2\Lambda(n)/n^{2.5} \leqslant \frac{4}{243} + \frac{10}{2187} = 0.0210333\ldots$ or $\phi(n) \geqslant 0.6292$. This yields an overall competitive ratio of not more than 1.123/0.6292, i.e., 1.7848. $\square$

A more refined analysis of $\Lambda(n)$ considers maximizing $\sum_i c_i^2 + \sum_j r_j^2$ all at once, instead of $\sum_i c_i^2$ and $\sum_j r_j^2$ separately, for a maximum value of $n(2\sqrt{n}+5) + \frac{n^2}{2\sqrt{n}+5}$. For $n \geqslant 81$, this yields $2\Lambda(n)/n^{2.5} \leqslant \frac{2}{243} + \frac{5}{2187} + \frac{2}{621} = 0.0137373\ldots$. As the resulting competitive ratio of 1.7643 is only very slightly better, we omit further details. If instead we rely on the unproven conjecture in [5] that $\frac{2c_{town}}{n^{2.5}} \approx \psi - \frac{0.410}{n}$, we get $\phi \geqslant 0.6451$, which corresponds to experimental evidence; the resulting competitive factor is 1.7406.

7. Lower bounds

We demonstrate that there are non-trivial lower bounds for a competitive factor. We start by considering the discrete online scenario for towns.

Theorem 15. *No online strategy can guarantee a competitive factor smaller than* $\frac{64}{\sqrt{5}^5} = 1.144866\ldots$.

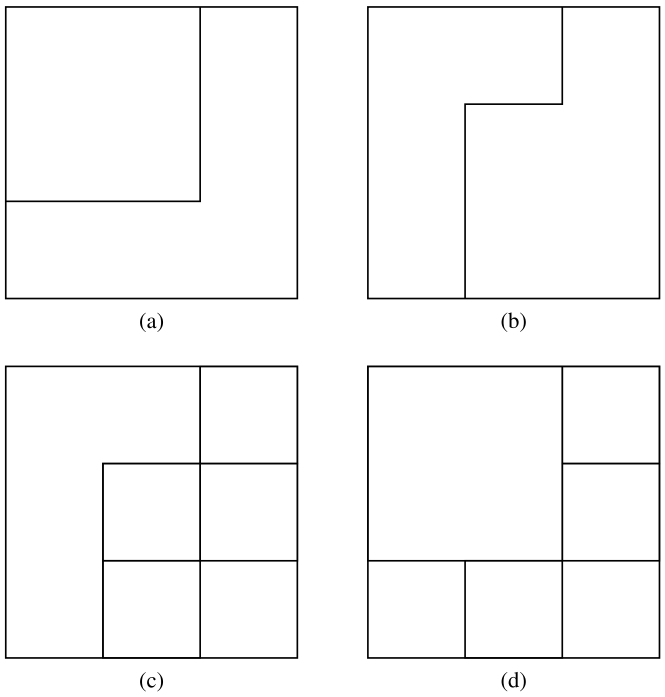

Fig. 7. The four cases considered in Theorem 15; the left column shows the choices by an algorithm, the right the corresponding optimal choices for the ensuing sequence.

Proof. Consider a 3×3 square, and let $n_1 = 4$; see Fig. 7. If (a) the strategy allocates a 2×2 square (for a total distance of 8), then $n_2 = 5$, and the resulting L-shape has a total distance of 20 and a ϕ-value of $40/5^{2.5} = 0.715541\ldots$. (b) Allocating the first town with an L-shape of total distance 10 results in $\phi = 20/32 = 0.625$, and the second with a total distance of 16, or $\phi = 32/5^{2.5} = 0.572433\ldots$.

If instead, (c) the first town is allocated different from a square, the total distance is at least 10, and $\phi \geqslant 20/32$; then (d) $n_2 = n_3 = n_4 = n_5 = n_6 = 1$, and an optimal strategy can allocate the first town as a 2×2 square, with $\phi = 0.5$. This bounds the competitive ratio, as claimed. □

For the case of continuous allocations, we claim the following.

Theorem 16. *There is $\delta > 0$, such that no online strategy can guarantee a competitive factor $1 + \delta$.*

Proof. Consider $n_1 = 1/2$, in combination with the two possible scenarios

(a) $n_2 = 1/2$;
(b) $n_2 = n_3 = \cdots = \varepsilon$.

In scenario (a), an adversary can assign two $(1 \times 1/2)$-rectangles, for a ϕ-value of $0.707\ldots$; in scenario (b), an adversary can assign all shapes as squares, for a ϕ-value of $0.666\ldots$. If the player chooses a square size $\sqrt{2}/2$ first, the adversary can choose scenario (a), causing the second allocation to be in L-shape with ϕ-value $\frac{2}{3}(7 - 4\sqrt{2}) = 0.895431\ldots$, as opposed to the optimal value of $0.707\ldots$. If the player chooses a $(1 \times 1/2)$-rectangle first, the adversary chooses scenario (b), for a ratio of $1.06066\ldots$. The existence of the claimed lower bound follows from continuity, as the ϕ-value changes continuously with continuous deformation of the involved shapes. □

The precise value arising from the scenarios in Theorem 16 is complicated. It can be obtained by computing the optimal intermediate value for the player that allows him to protect against both scenarios at once. For example, optimizing over the family of allocations shown in Fig. 8(c) yields a competitive ratio that is better than 1.06; however, the player may do even better by using curved boundaries. The involved computational effort for the resulting optimization problem promises to be at least as complicated as computing the "optimal shapes of a city", for which no closed-form solution is known, see [8,1].

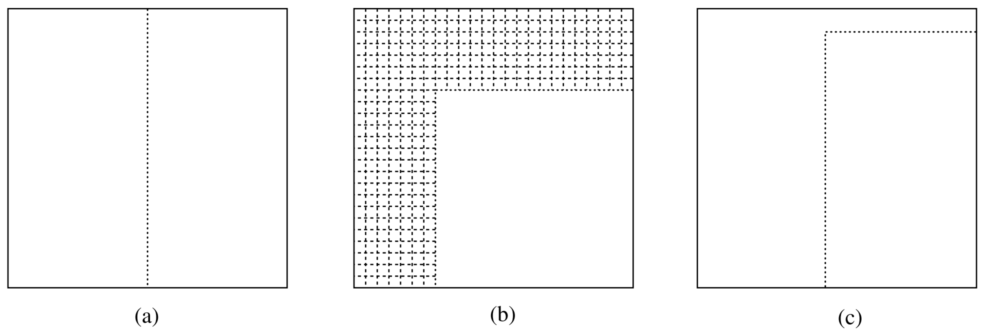

Fig. 8. The scenarios considered in Theorem 16, and a possible choice for the player.

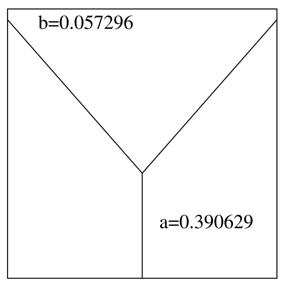

Fig. 9. A possible worst-case scenario for the offline problem.

8. Conclusions

We have established a number of results for the online shape allocation problem. In principle, further improvement could be achieved by replacing the computational results for level 3 (i.e., $n = 16, \dots, 64$) by level 4 (i.e., $n = 65, \dots, 256$). (Conversely, a simplified analysis with level 2, i.e., $n = 4, \dots, 16$; yields a worse factor of 3.6525.) However, the highest known optimal ϕ-values are for $n = 80$, obtained by using the $O(n^{7.5})$ algorithm of [5]. In any case, there is a threshold of 1.3504 for Hilbert-based strategies, which we believe to be tight: this is the ratio between the upper bound of 0.8768 for $n = 14$ (and for $n = 56, 224, \dots$) and the asymptotic lower bound of 0.650245; because asymptotically, continuous and discrete case converge, this also applies to the discrete case. Other open problems are to raise the lower bound of 1.144866 for the discrete case, and establish definitive values for the continuous case.

As noted in Section 3, we can eliminate the assumption of all n_i being multiples of some 2^{-R}, by making use of Lemma 11, and allocating a small round-off fraction to a fractional pixel maintains the same bounds. However, the formal aspects of describing the resulting allocation scheme become somewhat tedious and would require more space than seems appropriate.

The offline problem is interesting in itself: for given n_i, allocate disjoint regions of area n_i in a square, such that the maximum average Manhattan distance for each shape is minimized. As mentioned, there is some indication that this is an NP-hard problem; however, even relatively simple instances are prohibitively tricky to solve to optimality, making it hard to give a formal proof. Clearly, our online strategy provides a simple approximation algorithm; however, better factors should be possible by exploiting the a-priori information of knowing all n_i, e.g., by sorting them appropriately.

Another interesting open question for the offline scenario is the maximum optimal ϕ-value for any set $n_1, \dots, n_i$. A simple lower bound is $2/3 = 0.666\dots$, as that is the average distance of the whole square. A better lower bound is provided by dividing the square into two or three equal-sized parts. For the case $n_1 = n_2 = 1/2$, we can use symmetry and convexity to argue that an optimum can be obtained by a vertical split, yielding $\phi = \sqrt{2}/2 = 0.707$. We believe the global worst case is attained for $n_1 = n_2 = n_3 = 1/3$. Unfortunately, it is no longer possible to exploit only symmetry for arguing global optimality. Fig. 9 shows an allocation with $\phi = 0.718736\dots$ for all three regions.[1] We conjecture that this is the best solution for $n_1 = n_2 = n_3 = 1/3$, as well as the worst case for any optimal partition of the unit square.

[1] More precisely, the involved values can be expressed as $a = \frac{1}{108}(55 - \frac{791}{\theta} + \theta)$ and $\phi = \frac{(9\,602\,477 - 13\,416\sqrt{585\,705})\theta + (202\,679 + 204\sqrt{585\,705})\psi^2 + 82\,133\theta^3}{77\,760\sqrt{3}\theta}$ with $\theta := (-16\,253 + 36\sqrt{585\,705})^{1/3}$.

Acknowledgements

A preliminary extended abstract version summarizing results of this paper appears in WALCOM 2013 [6]. We thank Bettina Speckmann for pointing out Refs. [16] and [4], and several anonymous reviewers for helpful hints to improve the presentation of this paper. Work by Sándor P. Fekete was partially supported by Deutsche Forschungsgemeinschaft (DFG) under contract FE407/17-1 as part of the research group FOR 1800, Controlling Concurrent Change.

References

[1] C.M. Bender, M.A. Bender, E.D. Demaine, S.P. Fekete, What is the optimal shape of a city?, J. Phys. A 37 (1) (2004) 147–159.
[2] M.A. Bender, D.P. Bunde, E.D. Demaine, S.P. Fekete, V.J. Leung, H. Meijer, C.A. Phillips, Communication-aware processor allocation for supercomputers: finding point sets of small average distance, Algorithmica 50 (2) (2008) 279–298.
[3] H.-K. Dai, H.-C. Su, On the locality properties of space-filling curves, in: 14th International Symposium on Algorithms and Computation (ISAAC), 2003, pp. 385–394.
[4] M. de Berg, B. Speckmann, V. van der Weele, Treemaps with bounded aspect ratio, in: 22nd International Symposium on Algorithms and Computation (ISAAC), 2011, pp. 260–270.
[5] E.D. Demaine, S.P. Fekete, G. Rote, N. Schweer, D. Schymura, M. Zelke, Integer point sets minimizing average pairwise L1 distance: what is the optimal shape of a town?, Comput. Geom. 40 (2011) 82–94.
[6] S.P. Fekete, N. Schweer, J. Reinhardt, A competitive strategy for distance-aware online shape allocation, in: 7th International Workshop on Algorithms and Computation (WALCOM), 2013, pp. 41–52.
[7] C. Gotsman, M. Lindenbaum, On the metric properties of discrete space-filling curves, IEEE Trans. Image Process. 5 (5) (1996) 794–797.
[8] R.M. Karp, A.C. McKellar, C.K. Wong, Near-optimal solutions to a 2-dimensional placement problem, SIAM J. Comput. 4 (3) (1975) 271–286.
[9] S. Krumke, M. Marathe, H. Noltemeier, V. Radhakrishnan, S. Ravi, D. Rosenkrantz, Compact location problems, Theoret. Comput. Sci. 181 (2) (1997) 379–404.
[10] J.Y.-T. Leung, T.W. Tam, C.S. Wing, G.H. Young, F.Y. Chin, Packing squares into a square, J. Parallel Distrib. Comput. 10 (3) (1990) 271–275.
[11] V.J. Leung, E.M. Arkin, M.A. Bender, D.P. Bunde, J. Johnston, A. Lal, J.S.B. Mitchell, C.A. Phillips, S.S. Seiden, Processor allocation on Cplant: achieving general processor locality using one-dimensional allocation strategies, in: Proc. IEEE CLUSTER'02, 2002, pp. 296–304.
[12] R. Niedermeier, K. Reinhardt, P. Sanders, Towards optimal locality in mesh-indexings, Discrete Appl. Math. 117 (1–3) (2002) 211–237.
[13] H. Sagan, Space-Filling Curves, Springer, New York, 1994.
[14] N. Schweer, Algorithms for packing problems, PhD thesis, Braunschweig, 2010.
[15] R. Siromoney, K. Subramanian, Space-filling curves and infinite graphs, in: Graph Grammars and Their Application to Computer Science, in: Lecture Notes in Comput. Sci., vol. 153, Springer, Berlin, 1983, pp. 380–391.
[16] M. Wattenberg, A note on space-filling visualizations and space-filling curves, in: Proceedings of the IEEE Symposium on Information Visualization (INFOVIS), 2005, pp. 181–186.