BSc Thesis Applied Mathematics

Greedy algorithms for anchored rectangle packings

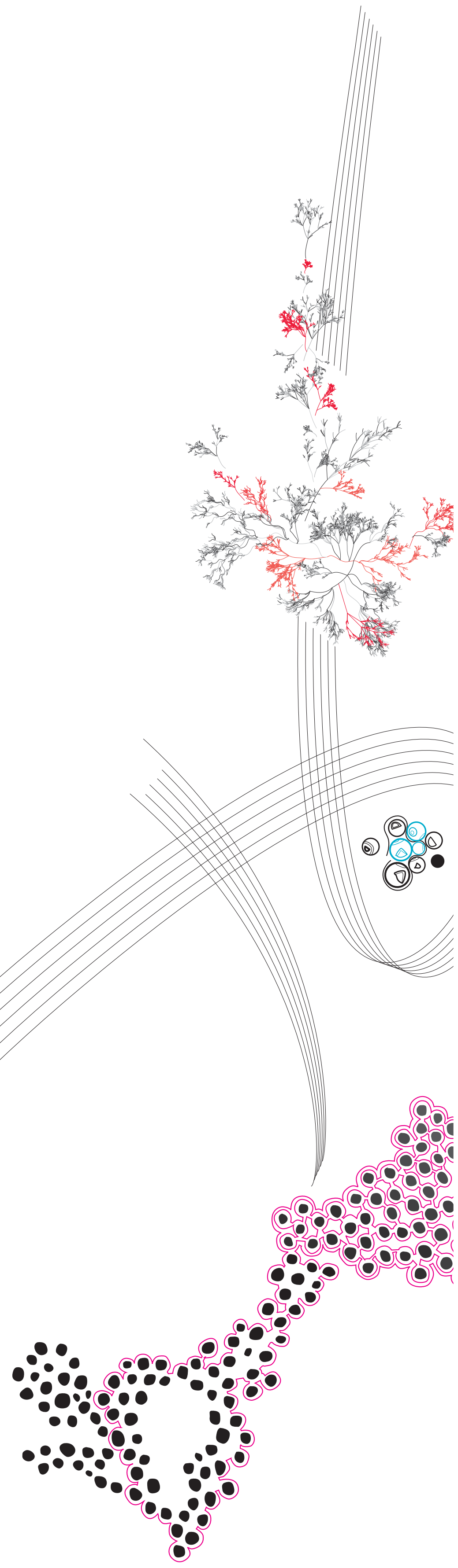
M.T. Maat

Supervisor: dr. R.P. Hoeksma

July, 2020

Department of Applied Mathematics
Faculty of Electrical Engineering,
Mathematics and Computer Science

UNIVERSITY OF TWENTE.



Preface

Greedy algorithms for anchored rectangle packings

M.T.Maat

July, 2020

Abstract

A lower-left anchored rectangle packing of a finite set of points S (including the origin) in the unit square is a set of axis-aligned rectangles in the unit square such that no two rectangles overlap, no points are in the interior of a rectangle and each rectangle has exactly one point of S as its lower left corner. A greedy algorithm to find such packings with large area was discovered before. It treats the points in a specific order. We derive principles for orderings for which this greedy algorithm yields a large area. We analyze the performance of a number of orderings empirically on a number of random point sets. Finally, we derive upper bounds for the worst case performance, and increase the best known lower bound on the worst case performance of an ordering to 0.9612.

Keywords: anchored rectangle, packing problem, greedy algorithm, computational geometry, approximation algorithm

1 Introduction

This thesis concerns a conjecture that was proposed over 50 years ago. A nice formulation of the problem is as follows (see Christ et al. [5]): Alice has baked a cake for her and Bob. It is a square cake, and Alice has put some raisins on top, of which one in the bottom left corner. Bob will cut the cake, but he has to follow some rules: he can only take rectangular axis-aligned pieces for himself, and all his pieces must have exactly one raisin, which must be at the lower left corner of each piece (and he cannot turn pieces). The conjecture states that Bob can always secure half of the cake for himself, independent of where Alice places the raisins. The choice of pieces for Bob we call a *(lower-left) anchored rectangle packing.* Although simple, the conjecture is still unsolved. In this thesis, we look at algorithms for Bob to secure as much 'cake' as possible, in particular we look at a greedy algorithm called *greedypacking* that chooses the largest possible rectangle at each point (raisin) in some order. The main research question is:

- Is there an easily described order of the points in the (lower-left) anchored rectangle packing problem, such that the greedypacking algorithm performs well?

In doing so, we state some principles that orderings should obey. Furthermore, we look at the optimal solutions for Bob. We propose a number of different orderings, and we compare the performance of these orderings empirically. Finally, we derive some upper bounds, and an improved lower bound for the worst case performance of greedypacking with some orderings.

2 Preliminaries

First, we state some definitions that will be used often in this thesis. Consider a set R of interior-disjoint rectangles in the unit square $U = [0,1]^2$, with sides parallel to the sides of U, and a finite set $S \subset U$. In this thesis, we will assume $(0,0) \in S$, unless stated otherwise (which is only for the dppacking algorithm), and we will always assume no two points share an x- or $y-$coordinate. We say a rectangle of R is *anchored* at a point p if p is the lower left corner of the rectangle, and there is no point of S in the interior of the rectangle. We denote the rectangle anchored at p by $r(p)$. We call p the *anchor* of $r(p)$. We call R an *anchored rectangle packing*[1] (ARP) of S if each rectangle in R is an anchored rectangle, and there is one anchored rectangle for each point. Define $N_S = |S|$, and denote by $A(R)$ the total area of the rectangles in R.
We say a point $p = (x_p, y_p)$ *dominates* a point $q = (x_q, y_q)$ if $x_p > x_q$, and $y_p > y_q$. We define the *dominance hull* D of a set $X \subset U$ by

$$D(X) = \{p \in U \mid p \text{ dominates } x \text{ for some } x \in X\}$$

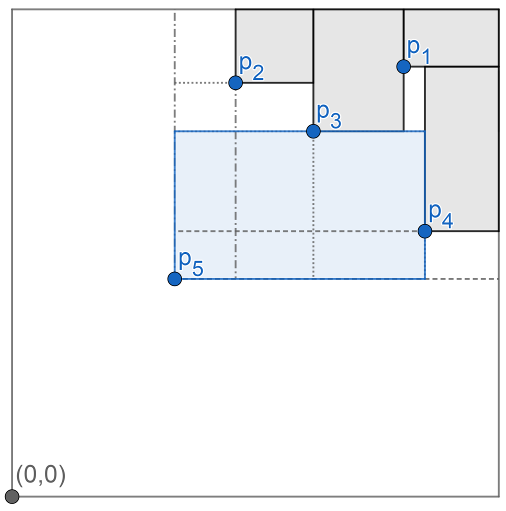


FIGURE 1: Example of a step in the greedypacking algorithm. In this case, p_5 is being treated, where p_1, p_2, p_3, p_4 have been treated before p_5, hence they have an anchored rectangle already (grey rectangles). Anchored rectangles cannot intersect, but can touch other rectangles, so the anchored rectangle with the largest possible area at p_5 will touch the grey rectangles. In this case, the blue rectangle has a larger area than the rectangles with dashed and dotted lines, so this will be the choice for $r(p_5)$.

In this thesis, the *greedypacking* algorithm is the following algorithm to find an anchored rectangle packing: treat the points of S in a specific order, and start with R empty. We denote an ordering by $\pi = (p_1, p_2, \ldots, p_{N_S})$. When a point is treated, find the maximum area rectangle anchored at that point, that is interior-disjoint with all other rectangles of R and all points of S, and add this to R (ties are decided arbitrarily). See Figure 1. When

[1] Sometimes this is called a 'lower-left anchored rectangle packing', as there are versions of this problem where the anchor can be on any corner, or on another place in the rectangle. In this article we will only consider the version with the anchor at the lower left corner, hence we use this name for simplicity.

all points are treated, R is an ARP. This algorithm can be implemented in time $\Theta(N_S^2)$ (Muller-Itten [9]). A *maximal* anchored rectangle packing is a rectangle packing whose area cannot be improved by changing one of the rectangles. Note that the greedypacking algorithm always yields a maximal ARP.

3 Related work

In this section we discuss some of the results that have been found before on this subject. The subject of anchored rectangle packings was first mentioned in a conference paper by Tutte [10] in 1969. In this paper, an open conjecture proposed by Allen Freedman was stated, which says that for each point set S, there exists an anchored rectangle packing of area at least $\frac{1}{2}$.
The problem appeared again, a long time later in a puzzle of IBM [8] and in a book of Peter Winkler [11], where the origin of the problem was unknown at first. After this, several results were found, setting several lower bounds on the area that can be covered.

First, Müller-Itten [9] presented the greedypacking algorithm in her master's thesis. Also, she presented a more complex algorithm for packing rectangles, and showed that this algorithm always finds an anchored rectangle packing R with $A(R) \geq \frac{1}{N_S}$. Finally, she showed that any maximal anchored rectangle packing can be constructed by the greedy-packing algorithm, which result we will use later.
Later, Christ et al. [5] improved this bound. They showed that, for sufficiently large r, the following holds: If there is a set S for which $A(R) \leq \frac{1}{r}$ for all anchored rectangle packings R of S, then $N_S \geq 2^{2^{\frac{r}{2}}}$. Reversing this statement would say $A(R) \geq \frac{1}{2\log_2 \log_2 N_s}$ for N_S large enough.

The first constant lower bound for the problem was established by Dumitrescu and Tóth [6]. In their paper, it was shown that an algorithm that partitions U into disjoint staircase-shaped areas, and picks the largest rectangle within each staircase, results in an anchored rectangle packing R with $A(R) \geq 0.09121$. We will improve this bound later. Furthermore, they showed that the greedypacking algorithm, if the points are treated from high to low in the ℓ_1-norm, performs at least as well as the staircase algorithm.
Some other results on lower-left anchored rectangle packing were found as well. Recently, Gadea Harder [7] proposed in his bachelor's thesis a dynamic programming algorithm, that computes the anchored rectangle packing with the largest possible area in exponential time.

Furthermore, the number of maximal anchored rectangle packings of a set S was shown to be at least $\Omega(\frac{4^{N_S}}{\sqrt{N_S}})$ and at most $\Theta(\frac{8^{N_S}}{N_S^{\frac{3}{2}}})$ in a paper by Balas et al. [2]. They also proved exponential upper and lower bounds for the related problem, where rectangles can be anchored in any corner point (instead of at its lower left corner).

On related problems where the rectangles can be anchored in different ways, Antoniadis et al. [1] showed that the related problem where the rectangles each have a point of S at their center is NP-hard. Moreover, they constructed for a given ϵ a polynomial-time algorithm that approximates the optimal solution by a factor $1 - \epsilon$, for the version of the anchored rectangle problem where each rectangle has a point in its center instead of its lower left corner.
Moreover, a paper from Balas et al.[3] describes the related problem where the anchors can be at any of the four corners of the rectangles in R, and the problem where R can only consist of squares instead of rectangles, and the combination of these two. It describes simple algorithms, and derives lower bounds for the performance of these algorithms.
Finally, the related problem where all points of S are on the boundary of U, and the

anchors can be on any corner of the rectangle, was studied by Biedl et al. [4]. It gives a polynomial-time algorithm that finds the best solution.
From earlier found results, we derive some principles for orderings in the greedypacking algorithm.

4 Some main principles

In this section we discuss some useful observations for ordering of points in a greedypacking algorithm. Some simple observations that were made earlier by Muller-Itten [9] are the following:

Observation 4.1. *Let L be the set of points at the top right of U (that is, the points p in S for which there is no other point of S that dominates p). The points of L can be dealt with easily: we can assume the rectangles of the points in L cover exactly their whole dominance hull, without limiting the area that can be covered by an anchored rectangle packing.*

Observation 4.2. *The rectangle $r((0,0))$ can be chosen independently from all the other points, as any rectangle anchored at the origin cannot affect the other choices and vice versa.*

Note that from Observation 4.1, we can derive that we can start an ordering with the points of S that are not dominated by any other points. Also, we can put them in any order, as it is not hard to see that their anchored rectangles cover their whole dominance hull for any order.
Another observation that we can make is the following theorem:

Theorem 4.3. *Let π be an ordering of the points of S, and let R_π be the anchored rectangle packing that is produced by the greedypacking algorithm, with the points ordered according to π. Then there exists an ordering π' of S such that:*

1. *The greedypacking algorithm with order π' yields R_π, for some choice of tie breaks.*
2. *For all points $p, q \in S$ where p dominates q, p comes before q in π'.*

Proof. First, note that R_π is a maximal anchored rectangle packing, by definition of the greedypacking algorithm. From Theorem 6.2 in Muller-Itten [9], Lemma 4.1 in Dumitrescu and Tóth [6] we know that any maximal anchored rectangle packing R defines a partial order $\prec_R$ on the points of S as follows: $q \prec_R p$ if p dominates q or if $r(p)$ is the one of the (at most) two rectangles that are hit by the two axis-aligned rays shot upwards and to the right respectively, coming from q. Note that $q \prec_R p$ is a necessary condition for $r(p)$ to affect the choice for $r(q)$. Now let $\prec_R^*$ be any linear extension of $\prec_R$, and let π' be the ordering that places the points from high to low according to $\prec_R^*$. Clearly, the greedypacking algorithm with order π' will yield R_π for some choice of tie breaks, since, for all points q, the rectangles that affect the choice of $r(q)$ are chosen before q is processed. Furthermore, by definition of the partial order, we have $q \prec_R p$ if p dominates q, hence π' also satisfies the second condition of the theorem. □

We refer to the second condition of Theorem 4.3 as the *dominance property*. We can now see that we only have to consider point orders that satisfy the dominance property. Intuitively, this means that point orders can be assumed to start with the point at the upper right corner and end with those at the lower left corner of U. Next, we will look at the best possible anchored rectangle packings, and derive more principles from these.

4.1 Optimal anchored rectangle packings

This section concerns an algorithm to find optimal anchored rectangle packings, which we define as follows: An *optimal anchored rectangle packing* of a set S is an anchored rectangle packing with the largest area over all possible anchored rectangle packings of S. Define an *optimal ordering* of S to be an ordering for which the greedypacking algorithm creates an optimal anchored rectangle packing of S. We know that for every maximal ARP, there is an ordering that reconstructs the ARP. Also, if an ARP is optimal, then it is also maximal, as choosing a rectangle with a larger area at one point will result in an ARP with a larger area. Hence an optimal ordering exists for any set S. Also, define for an ordering π of a set S and a point p the following: if $p \notin S$, their *concatenation* is (πp), which is an ordering of $S \cup \{p\}$, that starts with π and ends with p. If $p \in S$, define the *difference* $\pi \setminus p$, which is an ordering of $S \setminus \{p\}$ that is the same as π, but with p removed.

First, one important note is the following: sorting points based on any function $g : U \to \mathbb{R}$ can never guarantee that the order is optimal. This follows from the following theorem:

Theorem 4.4. *Let $p, q \in U$ be two distinct points such that none of the two dominates the other point. Then there exist finite point sets $S_1, S_2 \subset U$, with $\{p, q\} \subset S_1$ and $\{p, q\} \subset S_2$ such that:*

1. *For each optimal ordering π_1 of S_1, p comes before q.*
2. *For each optimal ordering π_2 of S_2, q comes before p.*

Proof. By symmetry, it suffices to prove that S_1 exists. W.l.o.g. we assume $x_p < x_q$, hence $y_p > y_q$. Consider for a sufficiently small $\epsilon > 0$ the point

$$v = (1 + \frac{1 - x_q}{1 - y_q}(y_p + \epsilon - 1) + \epsilon, y_p + \epsilon)$$

This point is chosen such that v is just left of the line through q and $(1, 1)$. Furthermore, define the set $S_1 = \{(0, 0), p, q, v\}$ (see Figure 2). As there are only two maximal anchored rectangle packings for S (resulting from orderings $(v, p, q, (0, 0))$ and $(v, q, p, (0, 0))$), we can easily see that all orderings where p comes before q are optimal, and all orderings with q before p are not. Hence we found a set S_1 with the desired property. □

From this theorem, it also immediately follows that for two orderings that are based on functions $g_1 : U \to \mathbb{R}$ and $g_2 : U \to \mathbb{R}$ that satisfy the dominance property, there are always point sets such that the first ordering performs better than the second, and the other way around. So no such ordering is always better than another one.

To be able to compute optimal ARP's, we define the *dppacking algorithm* (see also Gadea Harder [7]). It works as follows. For a point set S (not necessarily containing the origin), consider all possible subsets of S, and do the following for each subset $T \subseteq S$, from smallest to largest cardinality of T: for all points $t \in T$, assume t is the last point in the optimal ordering of T. Then retrieve the optimal ordering π_t for $T \setminus \{t\}$ (for $|T| = 1$, this is trivial, for $|T| \geq 2$, this has then been calculated before). Then apply the greedypacking algorithm with ordering $(\pi_t t)$ on T. Compare the results for all $t \in T$, find the result with the t that yields the largest area for T, and save this for T as the optimum (ties are broken arbitrarily). Then the result of the algorithm for the last set, $T = S$, is an optimal ordering for S. See Algorithm 1 for the pseudocode.

Note that the dppacking algorithm can be accelerated by the principles found in the previous chapter. Due to Observations 4.1, 4.2 we only have to consider all subsets of S

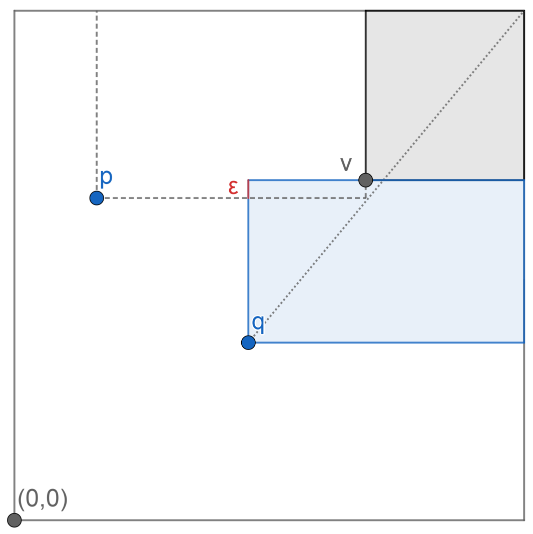


FIGURE 2: Point set S_1. If q comes before p, the blue rectangle is chosen at q, as v is just above the line through q and $(1,1)$ (dotted line). Then this rectangle is blocking a constant fraction (independent of ϵ) of the largest possible rectangle anchored at p (dashed lines). If p comes first, the rectangle anchored at q is reduced by less than ϵ, so an ordering with p before q yields a larger area for small enough ϵ.

Algorithm 1 Dppacking algorithm

```
1:  procedure DP($S$)
2:      treatedsubsets←subsets($1, S$)   ▷ Where subsets($i, S$) yields all subsets of size $i$ os $S$.
3:      for $p$ in $S$ do
4:          bestordersforsubset[$p$]←($p$)   ▷ bestorderforsubset is a dictionary with the best order for each subset.
5:      end for
6:      for $2 \leq i \leq size(S)$ do
7:          current_subsets←subsets($i, S$)
8:          for $T$ in current_subsets do
9:              bestvalue←0
10:             bestorder←[]
11:             for $t$ in $T$ do   ▷ Find the $t$ with the largest area if $t$ is last in the order.
12:                 if greedypacking((bestorderforsubset[$T \setminus \{t\}$],$t$))>bestvalue then
13:                     bestvalue←greedypacking((bestorderforsubset[$T \setminus \{t\}$],$t$))
14:                     bestorder←(bestorderforsubset[$T \setminus \{t\}$],$t$)
15:                 end if
16:             end for
17:         end for
18:     end for
19:     return bestorderforsubset[$S$]
20: end procedure
```

without the origin and we can start the algorithm with the points that are not dominated by any point in any order. Also, only $t \in T$ that do not dominate any point in T have to be considered due to Theorem 4.3.

Since the correctness of this algorithm has not been rigorously proved before, we prove it here. To prove that this algorithm works, we first prove the following lemma:

Lemma 4.5. *For each finite point set $S \subset U$ (that does not necessarily include the origin), there is a point $p_f \in S$ such that, for each optimal ordering π' of $S \backslash \{p_f\}$, the ordering $(\pi' p_f)$ is optimal for S.*

Proof. In the same way as in the proof of Theorem 4.3, we define a partial order $\prec_R$ on the points of S for any maximal anchored rectangle packing R (and therefore for any optimal ARP). Let π be an optimal ordering of S that satisfies the dominance property, and let R_π be its resulting ARP. Let p_f be the last point of π. We prove the lemma for this p_f. Furthermore, let π' be an optimal ordering for $S \setminus \{p_f\}$. Now, we assume that ordering $(\pi' p_f)$ is not optimal for S and derive a contradiction. Note that, since the greedypacking algorithm with ordering π' is optimal for $S \setminus \{p_f\}$, it covers at least as much area as with the ordering $\pi \setminus p_f$ on $S \setminus \{p_f\}$. Therefore $(\pi' p_f)$ must have a smaller rectangle $r(p_f)$ than the rectangle $r(p_f)$ in π (see Figure 3).

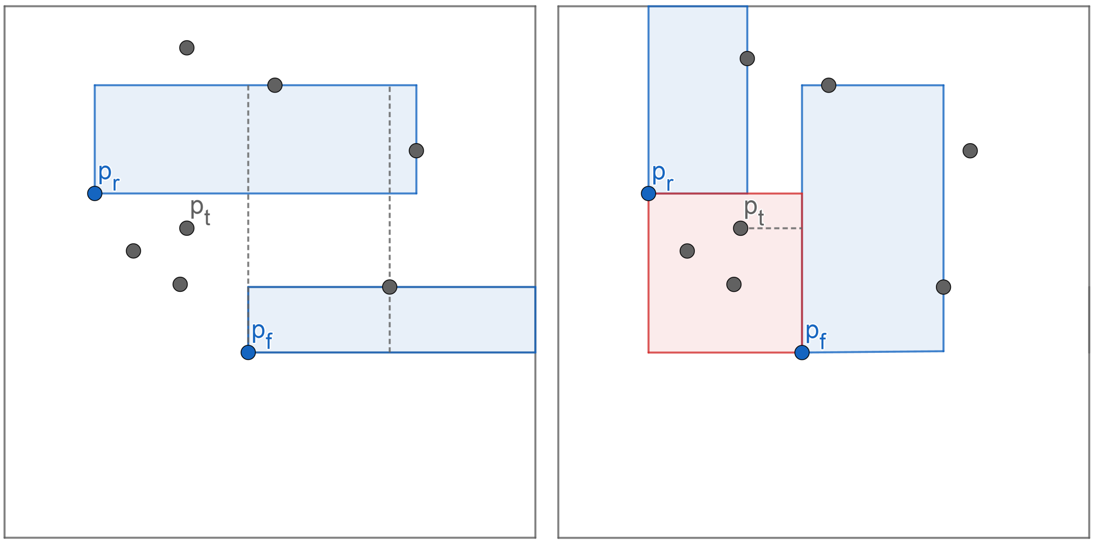


FIGURE 3: Left: part of the ARP with ordering $(\pi' p_f)$ on S. The largest possible rectangle at p_f in π (dashed lines) is blocked by $r(p_r)$. Right: part of the ARP with ordering π, where the rectangle with p_r and p_f as its corners (red) is nonempty. The largest possible rectangle can now be chosen for p_f, and we see that the ray to the right from p_t hits $r(p_f)$.

Therefore the $r(p_f)$ with the largest possible area in π cannot be chosen in the ordering $(\pi' p_f)$. We know the choice for a rectangle $r(p_f)$ is only restricted by the boundary of the unit square, by points that dominate p_f, and by rectangles that intersect one of the two axis-aligned rays going from p_f to the right and up (since anchored rectangles are only stopped by the boundary of U and other rectangles, and these other rectangles either dominate p_f or intersect a ray if part of the rectangle dominates p_f). Therefore, there must be a rectangle $r(p_r)$ that intersects such a ray of p_f in $(\pi' p_f)$ but not in π. W.l.o.g. it intersects the ray going up. Now, we look at $r(p_f)$ in π. Because this rectangle was restricted by $r(p_r)$ in π', we know that p_r must lie to the left of p_f w.r.t. the x-coordinate, and above p_f and below the top right corner of $r(p_f)$ in π with respect to its y-coordinate. Now define an axis-aligned rectangle ρ in U (not an anchored rectangle) with p_r as its top left corner and p_f as its bottom right corner (red rectangle in Figure 3). Note that there are no points directly under ρ as p_f is not dominating any point of S, by the dominance property of π. Suppose rectangle ρ has no points in its interior, then the horizontal ray to the right from p_r hits $r(p_f)$ first, as the horizontal line segment from p_r to the left boundary of $r(p_f)$ cannot intersect any other rectangles than $r(p_r)$, since since only anchored rectangles from points inside or directly below ρ could 'block' the ray. Therefore $p_r \prec_{R_\pi} p_f$, hence p_f cannot be the last point in the ordering π. However, this contradicts the definition of p_f. Likewise, if there are points in the rectangle, let p_t be the rightmost point not equal to p_f inside this rectangle, then we find in a similar way $p_t \prec_{R_\pi} p_f$, with again a contradiction. We conclude the assumption that (π', p_f) was not optimal for S cannot be true, hence the lemma holds for this point p_f. □

Theorem 4.6. *The dppacking algorithm yields an optimal ordering.*

Proof. We prove this by induction on the size of the set of points. For a point set of size 1, this is trivial. Now, as induction hypothesis assume the theorem holds for sets S of size $k \geq 1$. Then consider a point set S of size $k+1$, and consider all subsets $S \setminus \{p_f\}$ for all choices of $p_f \in S$. By the induction hypothesis, the algorithm calculates an optimal ordering π_{p_f} for all $S \setminus \{p_f\}$, and by Lemma 4.5, we know that there is a p_f for which $(\pi_{p_f} p_f)$ is optimal. Therefore, since the dppacking algorithm yields the ordering $(\pi_{p_f} p_f)$ which gives the largest area, we conclude that the result of the dynamix programming algorithm must be an optimal ordering of S. □

4.2 Simulation

The dppacking algorithm was implemented in a Python program (see Appendix A). The worst case time complexity of the used implementation of the dppacking algorithm is $\Theta(N_S^3 \cdot 2^{N_S})$. When there is one point dominating all other points, and of the other points except $(0,0)$, no two points dominate each other, then only the place in the order of $(0,0)$ and the point on the top right are determined by the extra assumptions. For the other points, 2^{N_S-2} subsets are considered, and for each subset T of size N_T, the greedypacking algorithm with time complexity $O(N_T^2)$ is run N_T times, and N_T is on average $\frac{1}{2}N_S$, yielding a runtime of $\Theta(N_S^3 \cdot 2^{N_S})$ (we cannot have a higher runtime, as there are at most 2^{N_S} subsets and $N_T \leq N_S$). For this reason, only small point sets were considered (for example, an optimal ordering for a set of size 20 can take up to two minutes to calculate). To gain insight into the structure of optimal orderings of point sets, two empirical experiments were performed.

The first experiment is very simple: for a number of different orderings according to a function of the x and y-coordinate (which satisfy the dominance property) and for a random ordering (where any possible ordering, also orderings not satisfying properties discussed before, has equal probability), test how many times these rules yield an optimal ordering on a large number of point sets. To incorporate round-off errors, it was assumed that an ordering is optimal if the area it yields is less than 10^{-13} from the optimum.

We assume then that the probability that this happens for a non-optimal ordering is negligible[2]. Results are in Table 2 in appendix B, for a simulation of 200000 point sets.

For the uniform distribution, $\|(x,y)\|_2$, $-\|(1-x,1-y)\|_0$, -$\|(1-x,1-y)\|_{-1}$ and $-\|(1-x,1-y)\|_{-2}$ have more optimal orderings than the others (the Z-score for the smallest difference, "$-\|(1-x,1-y)\|_{-2}$ has more optimal orderings than $\|(x,y)\|_1$" is 7.75, with a p-value of 0.0000). For the exponential distribution, this is similar, only $\|(x,y)\|_1$ is now close to the largest values (Z-score of the largest difference, "$\|(x,y)\|_2$ has more optimal orderings than $\|(x,y)\|_1$" is 2.99, with a p-value of 0.0014). This could also be explained by the fact that these orderings are similar if all points are close to the left and bottom edge of U.

Now for the second experiment, we define a random variable X with probability density function $f_X : [0,1] \to \mathbb{R}_{\geq 0}$, and a positive integer N. We define a function $F_{f_X,N} : U \to [1,N]$ as follows: We consider random point sets S (including the origin) of size N, where both the x and $y-$coordinate are distributed according to f_X (except for the origin). Furthermore, for each optimal ordering of S, we label the points from 1 to N, where the point that is first in the ordering gets 1, the second gets 2, etcetera. Denote the label as $l(p,\pi)$ for a point p and ordering π. For a point $p=(x,y) \in U$, we then define $F_{f_X,N}(p)$ as the expected value of the label of a point at p over all possible point sets containing p and optimal orderings (where each optimal ordering has equal probability):

$$F_{f_X,N}(p) = E[l(p,\pi) | p \in S \wedge \pi \text{ is an optimal ordering of S}]$$

The values of this function can be approximated empirically by dividing the unit square into small regions, and then generating random point sets of size N with coordinates according to f_X. Since tie-breaking for the dppacking algorithm occurs randomly, the average label value of points inside the small region approximates the value of $F_{f_X,N}$ for the points in the region. In Figure 4, the results of two simulations can be found. Results of experiments with different parameters yielded rather similar results for the uniform distribution. More results can be found in Appendix B.

In all of the experiments, the resulting function $F_{f_X,N}$ has its minimum at the origin and its maximum at $(1,1)$ (and high values around the top and right border). This is logical, because points at the right and top are likely to have no dominating points, hence they are likely to be treated first, and because points more to the bottom and left are dominated by more points. Moreover, in all the results with $X \sim \text{Uniform}(0,1)$, it seems that $F_{f_X,N}$ showed concave level curves from the left border of U to the bottom or from the top border to the right. The level curves were quite well approximated by the level curves of $\sqrt{(1.1-x)(1.1-y)}$, but it is not so clear why this is the case. When, however, the distribution of the generated points changed, the shape of the distribution also changed. For very skewed distributions of X, the level curves were straight lines or convex curves. See for example Figure 17 in Appendix B.

In conclusion, in the first experiment, the orderings according to functions with concave level curves ($\|(x,y)\|_2$, $-\|(1-x,1-y)\|_0$, $-\|(1-x,1-y)\|_{-1}$ and $-\|(1-x,1-y)\|_{-2}$) have a higher proportion of optimal orderings than the other orderings. This result is a bit different from the second experiment, where the more skewed distributions of points yielded sorting functions with rather convex level curves. Note, however, that the shape

[2]From Balas and Tóth [2] we know there are at most $\frac{1}{11}\binom{20}{10}2^{10} \approx 1.7 \times 10^7$ maximal ARP's. In simulations, it seemed that the area from greedypacking is mostly between 0.7 and 0.9. To give an estimate, if the areas of ARP's were uniformly distributed between 0.7 and 0.9, the probability of an area of an ARP being less than 10^{-13} from the optimum would be about 9×10^{-6}, not even considering the probability of choosing this ARP and the fact that often many ARP's have the same area.

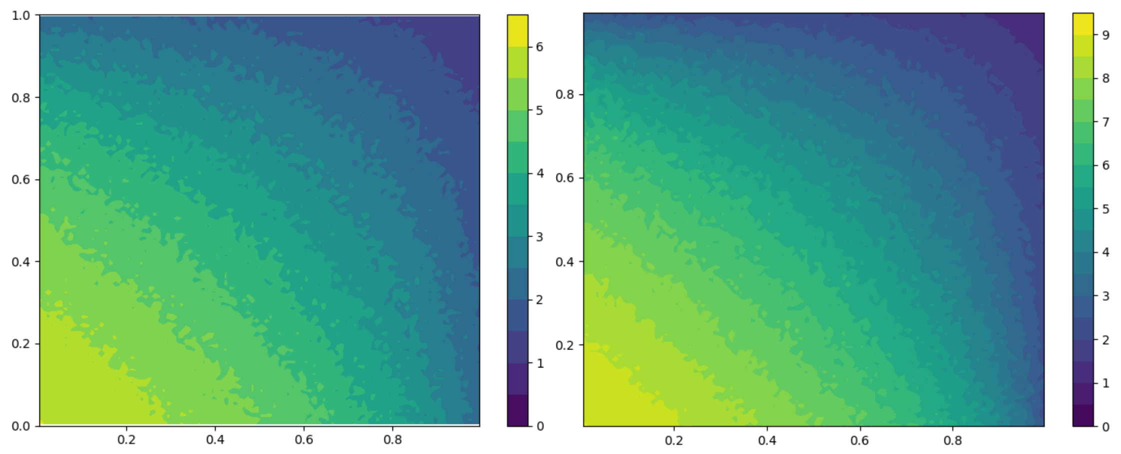


FIGURE 4: Contour plot of values of $F_{f_X,N}(p)$, for $X \sim Uniform(0,1)$, for $N = 7$ (left) and $N = 10$ (right), with division of U into 100×100 small squares, and 150000 generated point sets (label 0 means no data).

of the function $F_{f_X,N}$ is immediately dependent on the distribution of X, so this is not a contradiction to what was found on the first experiment, The convex shape of the level curves could also be explained by the observation that most of the points lie close to the bottom and left boundary of U. So it seems that the above mentioned distributions often yield optimal orderings compared to the other orderings.

5 Performance analysis

This section concerns performance analysis of different simple ordering rules. Both average case and worst case performance are considered.

Because Theorem 4.4 implies that good orderings might be dependent on relative locations of points, we introduce a new ordering strategy with three variations. It works as follows: the ordering starts with the points that are not dominated by any other points, and these are treated with the step of the greedypacking algorithm. When the rectangles for those points are determined, the following step is repeated:

- First, the set M is determined, consisting of the points that are not dominated by a point that is not treated yet. The next point p in the ordering is chosen from M according to some criterion. Then, p is treated in the step of the greedypacking algorithm.

For the three variantions, three different criteria to select a point are used:

1. *Euclidean ordering:* Let R_c be the set of rectangles that have already been chosen. Then choose the point p with the smallest Euclidean distance of p to the part of R_c that is in the quarter-plane to the right and above p.

2. *Area ordering:* Find a largest possible anchored rectangle over all possible rectangles anchored at any point in M. Choose as point p the anchor of this rectangle.

3. *Combined ordering:* Choose the point p with the smallest value of

$$\frac{\text{Euclidean distance to } R_c \text{ in the upper right quarter-plane}}{\|p\|_2}$$

In pseudocode:

Algorithm 2 Base algorithm for some orderings

```
procedure SomeOrderings($S$)
    ordering← NonDominatedPoints($S$)    ▷ Where NonDominatedPoints($X$) yields the points of $X$ that are not dominated by another point of $X$.
    $S$.remove(ordering)    ▷ Remove the treated points from $S$.
    while $S$ not empty do
        $M$ ← NonDominatedPoints($S$)
        minvalue← $\infty$
        for $q$ in $M$ do
            if Criterion($q$,ordering)<minvalue then    ▷ For some function Criterion(q,ordering).
                $p \leftarrow q$
                minvalue← Criterion($q$,ordering)
            end if
        end for
        ordering.append($p$)
        $S$.remove($p$)
    end while
    return ordering
end procedure
```

Finally, note that, as these orderings only consider points that are not dominated by a point that is not treated, they satisfy the dominance property.

5.1 Average case performance

First, we look at average case performance of different orderings. An analytic result is possible for one case. We define a function $g : U \to \mathbb{R}$, a probability density function $f : [0, 1] \to \mathbb{R}_{\geq 0}$ and a positive integer N_S. Consider an ARP of a set S that consists of N_S points, where all the x and y-coordinates of the points of S (except the origin) are pairwise independent and distributed according to f. Let the rectangles be decided by the greedypacking algorithm, with as ordering the ordering of points from high to low by value of g, where the points that are not dominated by any other point are always treated first. The *ordering expectation* $\mathrm{OE}_{f,N_S}[g]$ of g is defined as the expected area of the ARP given f and N_S. We assume the ordering by g satisfies the dominance property.

It is possible to find $OE_{f,N_S}[g]$ analytically for a given N_S by integrating over the areas of all different configurations. For $N = 4$, there are only three cases that make the difference between $OE_{f,N_S}[g]$ and $OE_{f,N_S}[min(x, y)]$, hence the difference can be calculated as follows:

Theorem 5.1. *Let $g : U \to \mathbb{R}$ be a continuous function with $g((x,y)) = g((y,x))$ for all $(x,y) \in U$, such that the ordering according to g satisfies the dominance property. Let $y = l_c(x)$ for constant c be the level curve defined by $g(x,y) = c$ in U. Finally, let $f_S(x_1,x_2,x_3,y_1,y_2,y_3)$ be the probability density function of the set $S = \{(0,0),(x_1,y_1),(x_2,y_2),(x_3,y_3)\}$, where all x,y-coordinates of $S \setminus \{(0,0)\}$ are pairwise independent and distributed according to the probability density function f. Then, we have*

$$\begin{aligned}
&\mathrm{OE}_{f,4}[g] - \mathrm{OE}_{f,4}[\min(x,y)] = \\
12 \quad \times \quad & \Bigg[\int_0^1 \int_0^{y_3} \int_{x_3}^{y_3} \int_0^{x_3} \int_{x_1}^{x_3} \int_{x_1}^{l_{g(x_1,y_1)}(x_2)} \Psi(S) dy_2 dx_2 dx_1 dy_1 dx_3 dy_3 \\
+ \quad & \int_0^1 \int_0^{y_3} \int_0^{x_3} \int_0^{y_1} \int_{x_1}^{y_1} \int_{x_1}^{l_{g(x_1,y_1)}(x_2)} \Psi(S) dy_2 dx_2 dx_1 dy_1 dx_3 dy_3 \\
+ \quad & \int_0^1 \int_0^{x_3} \int_0^{y_3} \int_0^{y_1} \int_{x_1}^{y_1} \int_{x_1}^{l_{g(x_1,y_1)}(x_2)} \Psi(S) dy_2 dx_2 dx_1 dy_1 dy_3 dx_3 \Bigg] \qquad (1) \\
\textit{If } \Psi(S) \quad = \quad & \Big((x_3 - x_2)(1 - y_1) - (y_3 - y_1)(1 - x_2) \Big) \cdot f_S(x_1,x_2,x_3,y_1,y_2,y_3)
\end{aligned}$$

Proof. We say $S = \{(0,0), p_1, p_2, p_3\}$, with $p_i = (x_i, y_i)$, $i = 1,2,3$. By symmetry, we can assume that $x_1 < x_2 < x_3$, and multiply the answer by 6 (the number of permutations) in the end. Now we will compare the ARP's produced by the orderings by g and by $\min(x,y)$. From Theorem 4.3 and Observations 4.1,4.2, it is easy to derive that an ordering is always optimal if not $y_2 < y_1 < y_3$, so this configuration is the only configuration we have to consider in comparing the difference[3]. In particular, we see that the only orderings that need to be considered are $(p_3, p_1, p_2, (0,0))$ and $(p_3, p_2, p_1, (0,0))$. We will consider the point sets where p_1 comes before p_2 when g is used, and p_2 before p_1 when $\min(x,y)$ is used. Note that the other way around is the same when the point set is mirrored along the line $x = y$ and points p_1 and p_2 switch their labels, hence we can just multiply by an extra factor 2 at the end.

Suppose $x_1 > y_1$. Since all level curves of g must be strictly decreasing (as its ordering satisfies the dominance property), we see that all points to the bottom right of p_1 have a lower value of $\min(x,y)$ than p_1 (as they have a lower y-value), but this contradicts the assumption that p_2 comes before p_1 for ordering by $\min(x,y)$. Hence $x_1 < y_1$.

Now we distinguish three cases (see also Figure 5):

1. $y_1 > x_3$: In all cases p_2 must be between the curves $y = l_{g(x_1,y_1)}(x)$ and $\min(x,y) = x_1$, to have the right ordering according to g and $\min(x,y)$. Since $y_1 > x_3$, the rightmost of the two intersections between $y = l_{g(x_1,y_1)}(x)$ and $min(x,y) = x_1$ has an $x-$coordinate greater than x_3, so p_2 lies in the area bounded by $x = x_1$, $y = x_1$, $y = x_3$ and $y = l_{g(x_1,y_1)}(x)$ (see Figure 5, left).

2. $y_1 < x_3$ and $x_3 < y_3$: If $y_1 < x_3$, then the rightmost intersection point between $y = l_{g(x_1,y_1)}(x)$ and $min(x,y) = x_1$ has an $x-$coordinate lower than x_3. Hence p_2 must lie in the area bounded by $x = x_1$, $y = x_1$, and $y = l_{g(x_1,y_1)}(x)$ (see Figure 5, right).

3. $y_1 < x_3$ and $x_3 > y_3$: This is similar to case 2. We consider $x_3 < y_3$ and $x_3 > y_3$ separately to be able to incorporate the condition $y_1 < \min(x_3, y_3)$ into an integral.

[3]For $y_1 < y_2 < y_3$, for $y_3 < y_2 < y_1$ and for $y_1 < y_3 < y_2$, any ordering is optimal. For $y_3 < y_1 < y_2$ and $y_2 < y_3 < y_1$, we have to treat the points that are not dominated first, and therefore all allowed orderings are optimal.

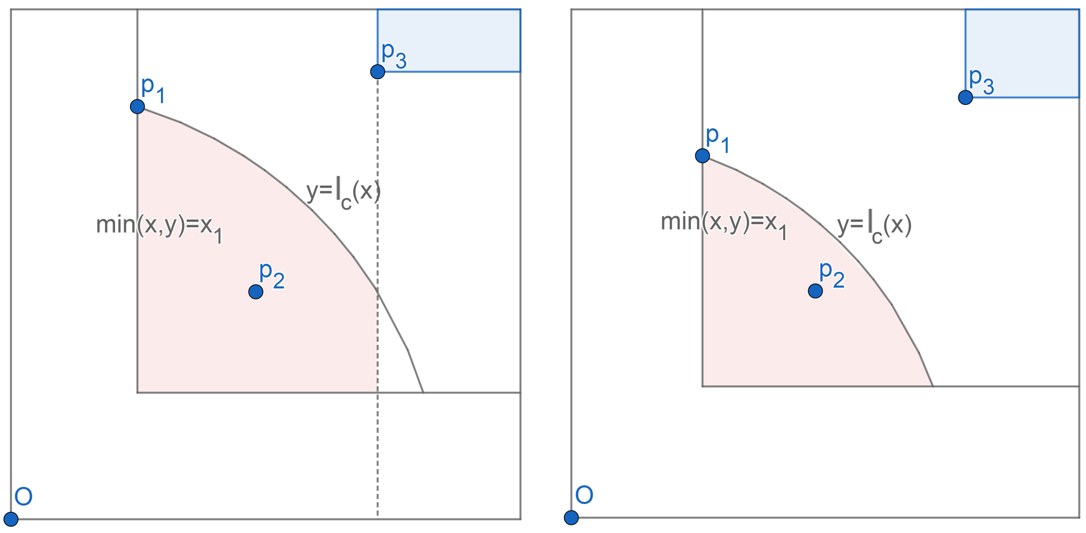


FIGURE 5: Left: the first case. p_2 lies between $y = l_{g(x_1,y_1)}(x)$ and $\min(x, y) = x_1$, and $x_2 < x_3$, so p_2 must be in the red shaded area. Right: case 2 and 3, point p_2 lies in the red shade area bounded by $\min(x, y) = x_1$ and $y = l_{g(x_1,y_1)}(x)$

Note that the difference between the anchored rectangle packing for g and of $\min(x, y)$ is given by $(x_3 - x_2)(1 - y_1) - (y_3 - y_1)(1 - x_2)$. Now for each case of these three, multiplying this formula by the probability density function of S, and integrating over the subset of S for which the case holds, yields the contribution for each case to the total value of $\mathrm{OE}_{f,4}[g] - \mathrm{OE}_{f,4}[\min(x, y)]$. These three cases correspond to the three integrals of (1). Finally, we multiply these integrals by 12 because of the symmetry assumptions that were made, and we get the difference $\mathrm{OE}_{f,4}[g] - \mathrm{OE}_{f,4}[\min(x, y)]$. □

For higher values of N_S, deriving such results in a similar way involves many cases, and they probably cannot be computed within reasonable time. Therefore, simulations were done on a large number of sets with N_S between 10 and 100. The uniform point distribution over U was used (except for the origin). The sets are still relatively small, because the implementation used for the algorithms described in this chapter is a bit slower ($\Theta(N_S^3)$ compared to $\Theta(N_S^2)$ for the greedypacking algorithm). For simplicity, only the best orderings from the simulations from last chapter were used. Results can be found in Table 3 in Appendix C.

At a significance level of 0.1%, four orderings yield the highest area at $N_S = 10$ (i.e. one is not proven better than the other at this significance level), namely $-\|(1 - x, 1 - y)\|_0$, $-\|(1 - x, 1 - y)\|_{-1}$, $\|(x, y)\|_1$ and $\|(x, y)\|_1$. Two orderings yield the highest area at this significance level at $N_S = 25$, namely $-\|(1 - x, 1 - y)\|_0$ and $\|(x, y)\|_1$. Finally, $\|(x, y)\|_1$ yields the highest area on average at $N_S = 50$ and $N_S = 100$ at this significance level. Furthermore, we see that all orderings have a high approximation ratio on average: about 0.98 to 0.99.

5.2 Worst case performance

Now we consider the worst case performance of different orderings (by "worst case" we mean the smallest area attained by using the ordering). We derive some upper bounds for the worst case performance and worst case approximation ratio.

First, we observe that if all points of S are close together on the line $x = y$, then the total area of any ordering approaches $\frac{1}{2}$, so that gives an upper bound of $\frac{1}{2}$ on the performance of any ordering. For the performance relative to the optimum, we derive some upper bounds on the approximation ratio.

Theorem 5.2. *The worst case approximation ratios for the greedypacking algorithm with an ordering according to* $\min(x, y)$ *from high to low, and for the area ordering are at most* $\frac{1}{2}$.

Proof. For all positive integers n and small enough real number $\epsilon > 0$, we construct a finite set $S_{n,\epsilon} \subset U$. Next, we show that when first $\epsilon \to 0$ and then $n \to \infty$, the area resulting from the ordering according to $\min(x, y)$ on $S_{n,\epsilon}$ and the area resulting from the area ordering on $S_{n,\epsilon} \subset U$ converge to $\frac{1}{2}$, while the area of the optimal ARP of $S_{n,\epsilon} \subset U$ converges to 1.

The set $S_{n,\epsilon}$ that is used is shaped like a staircase from $(0, 0)$ to $(1, 1)$, with steps of height and width about $\frac{1}{n+1}$, but with all points slightly perturbed. See also Figure 6. Define the point $v = (\frac{n}{n+1}, \frac{n}{n+1} + \epsilon^2)$. Let $p_i = (\frac{i-1}{n+1} + \epsilon^3, \frac{i}{n+1})$ and $q_i = (\frac{i-1}{n+1} + \epsilon, \frac{i-1}{n+1} + \epsilon^2)$ for $i = 1, 2, \ldots, n$. We will set $S_{n,\epsilon} = \{(0, 0), p_1, p_2, \ldots, p_n, q_1, q_2, \ldots, q_n, v\}$, and show that this set satisfies the claims. Because of dominating points, we see that v is first in both orderings, and $(0, 0)$ is last. We also see that p_i, q_i always come before p_j and q_j when $i > j$. Regarding the perturbations, note that $\min(x_{p_i}, y_{p_i}) = \frac{i-1}{n+1} + \epsilon^3 < \frac{i-1}{n+1} + \epsilon^2 = \min(x_{q_i}, y_{q_i})$. Moreover, the largest possible anchored rectangle at q_i has larger area than the possible anchored rectangles at p_i. So q_i comes before p_i for all i in both orderings. Taking into account all previous observations, we find that both the ordering according to $\min(x, y)$ and the area ordering are equal to $(v, q_n, p_n, q_{n-1}, p_{n-1}, \ldots, q_1, p_1, (0, 0))$.

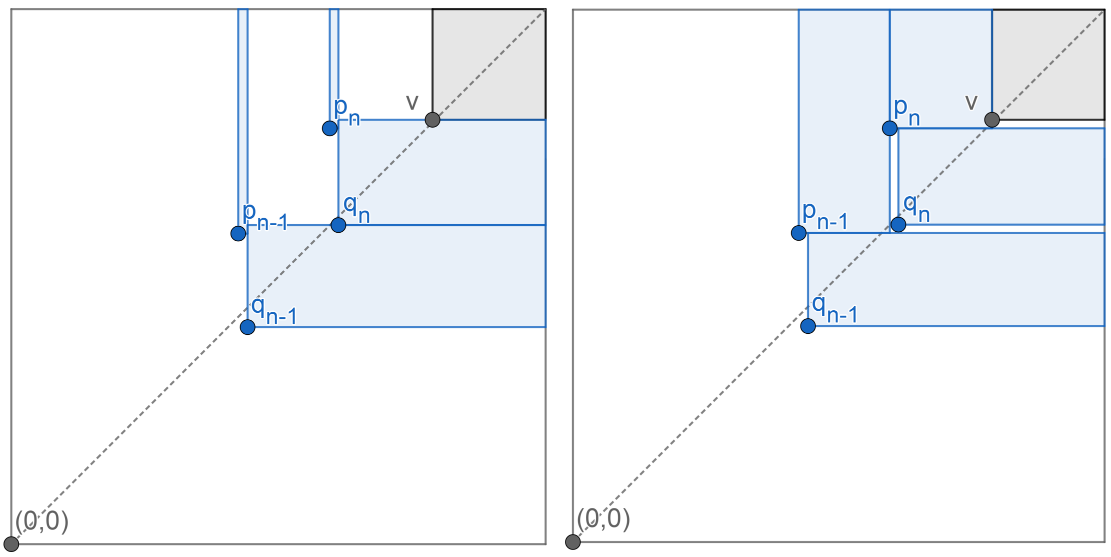


FIGURE 6: Left: part of the ARP from the ordering according to $\min(x, y)$ for point set $S_{n,\epsilon}$. We see $r(q_i)$ 'blocks' the point p_i. Right: part of the optimal ARP for $S_{n,\epsilon}$, almost the whole unit square is covered.

For the choice of $r(q_i)$ with $1 \leq i \leq n$, we find the following: there are two or three options for the rectangle, one that touches the top of U, one that touches the right of U, and for $i < n$, one where p_{i+1} is at the top edge of $r(q_i)$. The latter clearly has a much smaller area than the other two options. The first rectangle has an area of

$$\big((\frac{i}{n+1}+\epsilon^3)-(\frac{i-1}{n+1}+\epsilon)\big)\big(1-(\frac{i-1}{n+1}+\epsilon^2)\big)=\frac{n-i+2}{(n+1)^2}-\frac{(n-i+2)\epsilon+\epsilon^2}{n+1}+\epsilon^3-\epsilon^5$$

and the second one has an area of

$$\big(1-(\frac{i-1}{n+1}+\epsilon)\big)\big((\frac{i}{n+1}+\epsilon^2)-(\frac{i-1}{n+1}+\epsilon^2)\big)=\frac{n-i+2}{(n+1)^2}-\frac{\epsilon}{n+1}$$

We find that the rectangle touching the right edge of U is always the largest if $i \leq n$ and ϵ is small enough.
Finally, note that the top edge of $r(q_i)$ has y-coordinate $\frac{i}{n+1}+\epsilon^2$ for all i, so this edge has a higher y-coordinate than p_i, hence the rectangle at p_i cannot extend beyond q_i in the x-direction, and it can have a maximum width of $x_{q_i}-x_{p_i}=\epsilon-\epsilon^3$. See left half of Figure 6. We see that, as $\epsilon \to 0$, all areas reduce to 0, except the areas of $r(q_i)$ and $r(v)$, which have areas approximating $\frac{1}{(n+1)^2}, \frac{2}{(n+1)^2}, \dots, \frac{n+1}{(n+1)^2}$ (note that x_{p_1} and y_{q_1} approach 0, so the area of $r((0,0))$ goes to 0 as well). Hence the total area approaches $\frac{1}{2}+\frac{1}{2(n+1)}$, and we see that this approaches $\frac{1}{2}$ as $n \to \infty$.
On the other hand, note that we can choose the ordering of $(v, p_n, q_n, p_{n-1}, q_{n-1}, \dots, p_1, q_1, (0,0))$ instead. It is not hard to see that the resulting ARP covers the whole unit square except for $O(n)$ strips with width of $O(\epsilon)$ per strip. See also right half of Figure 6. Hence the area approaches 1 as $\epsilon \to 0$, this holds for all n. So the optimal ARP[4] has an area approaching 1 as $\epsilon \to 0, n \to \infty$ as well. Therefore, the worst case approximation ratio is at most $\frac{1}{2}$. $\square$

Theorem 5.3. *Let $g : U \to \mathbb{R}$ be a continuous function with $g((x,y)) = g((y,x))$ for all $(x,y) \in U$, and let that the ordering according to g from high to low satisfy the dominance property. Then the worst case approximation ratio for the greedypacking algorithm with ordering according to g is at most $\frac{3}{4}$.*

Proof. We construct a set $S_{n,\epsilon}$ for a positive integer n and a small real number $\epsilon > 0$. We show that, as first $n \to \infty$ and then $\epsilon \to 0$, the optimal area approaches 1 and the area for the above orderings goes to $\frac{3}{4}$. We define $S_{n,\epsilon}$ recursively. Let $p_1 = (\epsilon^2, \epsilon-\epsilon^3)$, let $q_1 = (\epsilon, \epsilon^2+\epsilon^3)$, $v_1 = (2\epsilon, 2\epsilon)$, $w_1 = (2\epsilon-\epsilon^2, \epsilon)$, and let $S_{1,\epsilon} = \{(0,0), p_1, q_1, w_1, v_1\}$. See also left side of Figure 7. Before defining the recursive relation, we first look at $S_{1,\epsilon}$. We see that v_1 dominates all other points, and w_1 dominates all other points except v_1. Furthermore, the mirror image of p_1 in the line $x = y$ is $p_1' = (\epsilon-\epsilon^3, \epsilon^2)$, and this is dominated by q_1. Since g is symmetric around the line $x = y$ and satisfies the dominance property, q_1 comes before p_1 in the ordering by g. So the ordering by g gives the ordering $(v_1, w_1, q_1, p_1, (0,0))$. Now we look at the options for choosing a rectangle $r(q_1)$. There are two options: one rectangle that touches the right edge of U, and one that touches the top edge of U. The first option has an area of

$$\big(1-\epsilon\big)\big(\epsilon-(\epsilon^2+\epsilon^3)\big)=\epsilon-2\epsilon^2+\epsilon^4$$

and the second one has an area of

$$\big((2\epsilon-\epsilon^2)-\epsilon\big)\big(1-(\epsilon^2+\epsilon^3)\big)=\epsilon-\epsilon^2-\epsilon^3+\epsilon^5$$

[4]In fact, from Theorem 4.3, we can derive that this ordering is optimal. We know that, to find the optimum, taking into account the point dominating each other, we find that we only have to make n independent choices for the order, namely between p_i and q_i, for all i independently. And clearly treating p_i before q_i always yields a larger area.

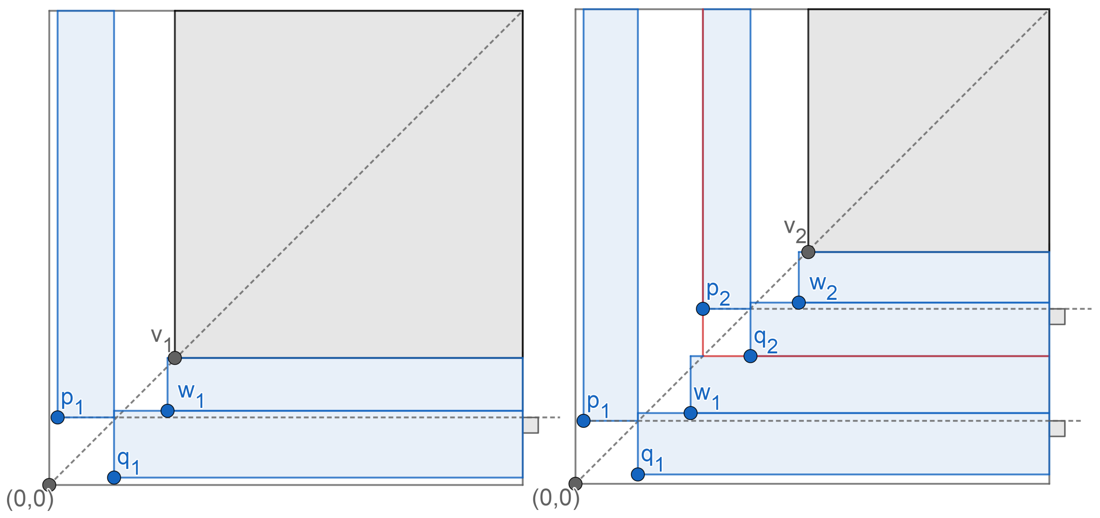


FIGURE 7: Left: the set $S_{1,\epsilon}$, with the ARP resulting from the ordering according to g (only $r((0,0))$ is not drawn). Rectangle $r(q_1)$ 'blocks' the point p_1. Right: set $S_{2,\epsilon}$, which is made by 'pasting' a copy of $S_{1,\epsilon}$ onto $r(v_1)$ in $S_{1,0.1}$, and deleting the two overlapping points. The choice of anchored rectangles is independent for the two 'L-shapes', which are separated by the red lines.

so we see that the first one has a larger area. Note that, with this choice for $r(q_1)$, the choices for all anchored rectangles are fixed, (except for $r(0,0)$, but all three possible rectangles are of area $O(\epsilon^2)$). We cover $1-\epsilon+O(\epsilon^2)$ of the unit square with anchored rectangles. In particular, we cover $3\epsilon+O(\epsilon^2)$ of the $4\epsilon+O(\epsilon^2)$-sized L-shape $U\setminus r(v)$, so we cover $\frac{3}{4}+O(\epsilon)$ of $U\setminus r(v)$.
Now we define the recursion, extending the principle from $S_{1,\epsilon}$. Consider $S_{n,\epsilon}$, then we get $S_{n+1,\epsilon}$ from it as follows: Take $S_{1,\epsilon}$, and map it onto U with the transformation $T : U \to [x_{v_n}, 1] \times [y_{v_n}, 1]$ given by $T((x,y)) = (x_{v_n} + \frac{x}{1-x_{v_n}}, y_{v_n} + \frac{y}{1-y_{v_n}})$. Delete the point v_n, and delete the point that was $(0,0)$ in $S_{1,\epsilon}$ (which was mapped onto v_n). Relabel the points that were labeled p_1, q_1, w_1, v_1 in $S_{1,\epsilon}$ to $p_{n+1}, q_{n+1}, w_{n+1}, v_{n+1}$, respectively. The resulting set is $S_{n+1,\epsilon}$. See right side of Figure 7. Now we can easily see that, for the steps in this recursion, the choice of rectangles of p_i, q_i, w_i is similar to those for p_1, q_1, w_1 in $S_{1,\epsilon}$, because p_i, q_i, w_i dominate p_j, q_j, w_j if $i > j$, so the choices are independent. The only difference in area is caused by replacing v_i by two points close to each other in the steps of the recursion. This creates a new possible rectangle, which is very small compared to the largest option, and increases the area of the other two options by $O(\epsilon^2)$, so the possible area for $r(w_i)$ changes with $O(\epsilon^2)$. Now, if we divide for the set $S_{n,\epsilon}$ the polygon $U\setminus r(v_n)$ into $L-shapes$ (see right part of Figure 7), we see that a fraction $\frac{3}{4}+O(\epsilon)$ of $U\setminus r(v_n)$ is covered by anchored rectangles. Since $\lim_{n\to\infty} \text{area}(r(v_n)) = \lim_{n\to\infty}(1-2\epsilon)^{2n} = 0$, we conclude that a fraction $\frac{3}{4}+O(\epsilon)$ is covered when $n\to\infty$. Letting then $\epsilon\to 0$, we see that a fraction $\frac{3}{4}$ is covered with the ordering according to g. Finally, using the ordering $(v_1, w_1, p_1, q_1, (0,0))$ on $S_{1,\epsilon}$ clearly covers a fraction $1+O(\epsilon)$ of $U\setminus r(v_1)$, hence the ordering $(v_n, w_n, p_n, q_n, \ldots, w_1, p_1, q_1, (0,0))$ of $S_{n,\epsilon}$ covers a fraction $1+O(\epsilon)$ of $U\setminus r(v_n)$. Similar to Theorem 5.2, we see that this ordering is optimal. Finally we derive that, as first $n\to\infty$ for a set $S_{n,\epsilon}$ and then $\epsilon\to 0$, we cover the whole unit square with the optimal ordering, since $1+O(\epsilon)$ of the L-shapes is covered. Therefore, the worst case approximation ratio is at most $\frac{3}{4}$ □

Theorem 5.4. *The worst case approximation ratio of the Euclidean ordering and the combined ordering is at most* $1 - \frac{\sqrt{2}}{4} \approx 0.646$.

Proof. We prove this in a similar way as Theorem 5.3. We construct a set $S_{n,\delta,\epsilon}$ for $n \in \mathbb{N}$ and for $\delta > 0$ and $\epsilon > 0$ small real numbers. Assume $\epsilon \ll \delta$. We show that, as first $n \to \infty$, then $\epsilon \to 0$, and then $\delta \to 0$, the ratio between the ARP area of these orderings compared to the optimal area approaches $1 - \frac{\sqrt{2}}{4}$.

The sets $S_{n,\delta,\epsilon}$ are defined recursively, with the same recursive step as in the proof of Theorem 5.3 (transforming $S_{1,\delta,\epsilon}$ onto $S_{n,\delta,\epsilon}$ with transformation T, and then deleting v_n and the origin from S_{1,δ_ϵ}), but with a different set for $n = 1$. Define $p_1 = (\epsilon^2, \epsilon)$, $q_1 = (\epsilon(\sqrt{2}-1)(1+\delta) + \epsilon^2, \epsilon^2)$, $w_1 = (\epsilon\sqrt{2} - \epsilon^2, \epsilon + \epsilon^2)$, and $v_1 = (\epsilon\sqrt{2}, \epsilon\sqrt{2})$, and let, again, $S_{1,\delta,\epsilon} = \{(0,0), p_1, q_1, w_1, v_1\}$ (see Figure 8). Let $S_{n,\delta,\epsilon}$ be derived from $S_{n-1,\delta,\epsilon}$ with a similar recursive step as in the proof of Theorem 5.3.

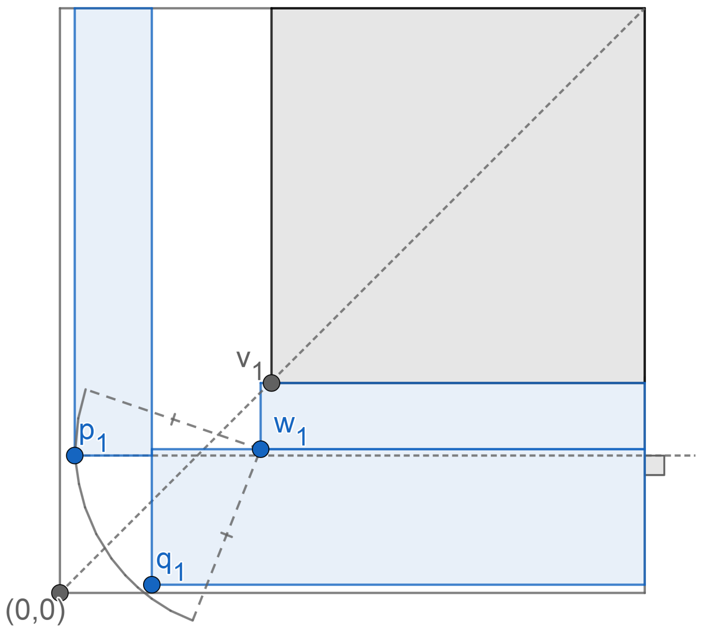


FIGURE 8: The set $S_{1,\delta,\epsilon}$, with the ARP resulting from the ordering according to the Euclidean ordering. The points are chosen such that q_1 is just inside the circle through p_1 with center w_1, and such that the anchored rectangle at q_1 'blocks' the point p_1.

Note that, because the algorithms satisfy the dominance property, the order of all points is already fixed, except for the choice between p_i and q_i, for $i = 1, 2, \ldots, n$. Clearly, the distance from p_i and q_i to the set of rectangles chosen before $r(p_i)$ and $r(q_i)$ is equal to the distance between p_i and w_i and between q_i and w_i, respectively. The distance between p_1 and w_1 is given by

$$\sqrt{\left((\epsilon\sqrt{2} - \epsilon^2) - \epsilon^2\right)^2 + \left((\epsilon + \epsilon^2) - \epsilon\right)^2} = \sqrt{2\epsilon^2 - 4\sqrt{2}\epsilon^3 + 5\epsilon^4} \tag{2}$$

and the distance between q_1 and w_1 is

$$\begin{aligned} &\sqrt{\left((\epsilon\sqrt{2} - \epsilon^2) - (\epsilon(\sqrt{2}-1)(1+\delta) + \epsilon^2)\right)^2 + \left((\epsilon + \epsilon^2) - \epsilon^2\right)^2} \\ = \; &\sqrt{\left(2 - 2\delta(\sqrt{2}-1) + \delta^2(\sqrt{2}-1)^2\right)\epsilon^2} \end{aligned} \tag{3}$$

We can see that $\frac{\text{distance}(q_1,w_1)}{\text{distance}(p_1,w_1)} = \sqrt{1-\delta(\sqrt{2}-1)+\frac{1}{2}\delta^2(\sqrt{2}-1)^2} + O(\epsilon)$. We also see that, since the points p_i, q_i, w_i are just scaled versions of p_1, q_1, w_1, we find that also $\frac{\text{distance}(q_i,w_i)}{\text{distance}(p_i,w_i)} = \sqrt{1-\delta(\sqrt{2}-1)+\frac{1}{2}\delta^2(\sqrt{2}-1)^2} + O(\epsilon)$. This implies that q_i always comes before p_i in the Euclidean ordering. We find for $S_{1,\delta,\epsilon}$ the following: the total area of the L-shape $U \setminus r(v_1)$ is $2\sqrt{2}\epsilon + O(\epsilon^2)$, and the covered area if q_1 comes before p_1 is $\epsilon(2\sqrt{2}-1) + O(\delta) + O(\epsilon^2)$, and if p_1 comes before q_1, it is $2\sqrt{2}\epsilon + O(\epsilon^2)$. Therefore, if p_i comes before q_i, we fill up a fraction $1 + O(\epsilon)$ of the L-shape containing p_1, q_1 with the ARP, and if q_1 comes before p_1, we cover a fraction $1 - \frac{\sqrt{2}}{4} + O(\delta) + O(\epsilon)$ of the L-shape. Since the shapes of all L-shapes are the same, this holds for all L-shapes. Now, since q_i comes before p_i in the Euclidean ordering, we see that, letting $n \to \infty$, a fraction $1 - \frac{\sqrt{2}}{4} + O(\delta) + O(\epsilon)$ of the optimal area is covered in the Euclidean ordering, hence letting first $\epsilon \to 0$ and then $\delta \to 0$, this proves the theorem for the Euclidean ordering.

Now to prove the theorem for the combined ordering, define $i_{\delta,\epsilon} \in \mathbb{N}$ to be the smallest natural number such that $\|p_{i_{\delta,\epsilon}}\|_2 > -\frac{2\epsilon}{\sqrt{1-\delta(\sqrt{2}-1)}-1}$ and $\|q_{i_{\delta,\epsilon}}\|_2 > -\frac{2\epsilon}{\sqrt{1-\delta(\sqrt{2}-1)}-1}$. Assume $n \geq i_{\delta,\epsilon}$. We know the distance between p_i and q_i is clearly always less than 2ϵ (their x and y-coordinates are at most ϵ apart), so the triangle inequality says for all $i \geq i_{\delta,\epsilon}$ that $\|q_i\|_2 + 2\epsilon \geq \|p_i\|_2$, hence for all $i \geq i_{\delta,\epsilon}$, by dividing by $\|p_i\|_2$:

$$\frac{\|q_i\|_2}{\|p_i\|_2} \geq 1 - \frac{2\epsilon}{\|p_i\|_2} > 1 - \frac{2\epsilon}{-\frac{2\epsilon}{\sqrt{1-\delta(\sqrt{2}-1)}-1}} = \sqrt{1-\delta(\sqrt{2}-1)} \tag{4}$$

$$\begin{aligned}\frac{\frac{\text{distance}(q_i,w_i)}{\|q_i\|_2}}{\frac{\text{distance}(p_i,w_i)}{\|p_i\|_2}} &= \frac{\text{distance}(q_i,w_i)}{\text{distance}(p_i,w_i)} \cdot \frac{\|p_i\|_2}{\|q_i\|_2} \\ &< \left(\sqrt{1-\delta(\sqrt{2}-1)+\frac{1}{2}\delta^2(\sqrt{2}-1)^2} + O(\epsilon)\right)\left(\frac{1}{\sqrt{1-\delta(\sqrt{2}-1)}}\right) > 1 \end{aligned} \tag{5}$$

Now from (5), we see that, from a distance more than $-\frac{2\epsilon}{\sqrt{1-\delta(\sqrt{2}-1)}-1}$ from the origin, the point q_i is treated before p_i, hence we cover a fraction $1 - \frac{\sqrt{2}}{4} + O(\delta) + O(\epsilon)$ of the L-shapes further than $-\frac{2\epsilon}{\sqrt{1-\delta(\sqrt{2}-1)}-1}$ away from the origin as $n \to \infty$. Then, letting first $n \to \infty$ and then $\epsilon \to 0$, we see $-\frac{2\epsilon}{\sqrt{1-\delta(\sqrt{2}-1)}-1} \to 0$, hence the area of the part of which $1 - \frac{\sqrt{2}}{4} + O(\delta) + O(\epsilon)$ is covered approaches 1, so the approximation ratio approaches $1 - \frac{\sqrt{2}}{4} + O(\delta)$ as $\epsilon \to 0$. Finally, letting $\delta \to 0$ (since $\epsilon \ll \delta$, we can do this after $\epsilon \to 0$), we find what we wanted to prove about the combined ordering, as the approximation ratio goes to $1 - \frac{\sqrt{2}}{4}$. □

The worst case performances of the considered algorithms in the simulations are in Table 4 in Appendix C. All worst case sets found had areas higher than the upper bounds proven in this section compared to the optimum, and larger than $\frac{1}{2}$ in absolute area.

5.3 Lower bound for worst case approximation

In this part, we prove a lower bound of 0.09612 for the performance of the ordering according to $\|(x,y)\|_1$. We do this by making two slight improvements on the analysis done by Dumitrescu and Tóth [6], which yields a lower bound of 0.9121. We introduce a variable α, and change a right-angled trapezoid into a parallelogram. This lower bound also implies that there exists an anchored rectangle packing of area 0.9612 for any point set S with $(0,0) \in S$.

First of all, we use the *tilepacking algorithm* to prove the lower bound. The tilepacking algorithm works as follows: treat the points according to their value of $\|(x,y)\|_1$, from high to low. When a point p_i is treated, shoot two axis-aligned rays, one upwards and one to the right, and call the staircase-like polygon that these rays create tile t_i, which is disjoint from the other tiles. Within each tile, choose the largest anchored rectangle in each tile. See left half of Figure 9. Then we use the following lemma (which is equivalent to Lemma 2.1 of [6]):

Lemma 5.5. *The greedypacking algorithm according to $\|(x,y)\|_1$ always covers at least as much area as the tilepacking algorithm with the same ordering.*

From this we conclude that it suffices to prove the lower bound for the tilepacking algorithm.

The main idea of the proof is that the area of the tiles that only allow relatively small anchored rectangles (called β-tiles) is bounded by a multiple of the area of a certain parallelogram associated with each β-tile. Also, the total area of the parallelograms of all these β-tiles is bounded, hence we get an upper bound on the total area of all β-tiles, which is less than 1. Finally, by using the integral equation from Theorem 3.1 in [6], we get the lower bound.

First, some definitions. We introduce a constant $\alpha > 0$, which we later optimize over (this is the first improvement, as originally, $\alpha = 1$ in [6]). Furthermore, we call a tile t_i a β-tile for $\beta \geq 3 + 2\alpha$ if the largest possible anchored rectangle contained in t_i has an area of at most $\frac{1}{\beta} \cdot \text{area}(t_i)$. Let the *right tip* of such a tile be the smallest axis-aligned polygon that is created by drawing a vertical (axis-aligned) line through a point of S that is a corner of t_i, and has area at least $\frac{\alpha}{\beta} \cdot \text{area}(t_i)$ (see Figure 9). Similarly, let the *upper tip* be defined as the smallest axis-aligned polygon that is created by drawing a horizontal line through a point of S that is a corner of t_i, and has area at least $\frac{\alpha}{\beta} \cdot \text{area}(t_i)$. Also, let the remaining part of t_i without its upper and right tip be its *main body* t_i'.

Let a_i and b_i be the bottom and left edge of t_i, respectively. Let a_i' be the bottom edge of the main body t_i', and let b_i' be the left edge of t_i'. We call a $\beta-$tile t_i *wide* if $|a_i'| \geq |b_i'|$, and we call it *tall* if $|a_i'| < |b_i'|$. Let Δ_i be the isosceles right triangle with the bottom edge of t_i as its top edge, and with the $90°$ angle at the bottom right point of t_i. Let $0 < \lambda < \alpha$, and let A_i be the parallelogram with one side on the left edge of Δ_i, with a_i' as one other edge, and with a height orthogonal to a_i' of $\lambda|a_i'|$. This is the second improvement, as A_i was originally a right-angled trapezoid in [6]. Now we show that the area of a wide β-tile t_i is limited by a multiple of the area of A_i:

Lemma 5.6. *The area of a wide β-tile t_i is at most $\frac{\beta}{\lambda e^{\beta-3-2\alpha}}$ area(A_i).*

Proof. First of all, we show that the area of the right tip is less than $\frac{1+\alpha}{\beta}$ area(t_i). If it had a larger area, we could move its left boundary one point (one 'step' on the staircase) to the right, and, as the difference is a small rectangle, which is part of some rectangle anchored at p_i, the difference in area is less than $\frac{1}{\beta}$ area(t_i), Therefore the new right tip would also have an area of more than $\frac{\alpha}{\beta}$ area(t_i), which is a contradiction. Similarly, the area of the upper tip is less than $\frac{1+\alpha}{\beta}$ area(t_i). Therefore, the area of the main body t_i' is greater than $\frac{\beta-2-2\alpha}{\beta}$ area(t_i), which is at least $\frac{1}{\beta}$ area(t_i), as $\beta \geq 3 + 2\alpha$. This implies that the right and upper tip do not overlap, as otherwise the main body would be a rectangle that is part of an anchored rectangle, which would have area at most $\frac{1}{\beta}$ area(t_i). Furthermore,

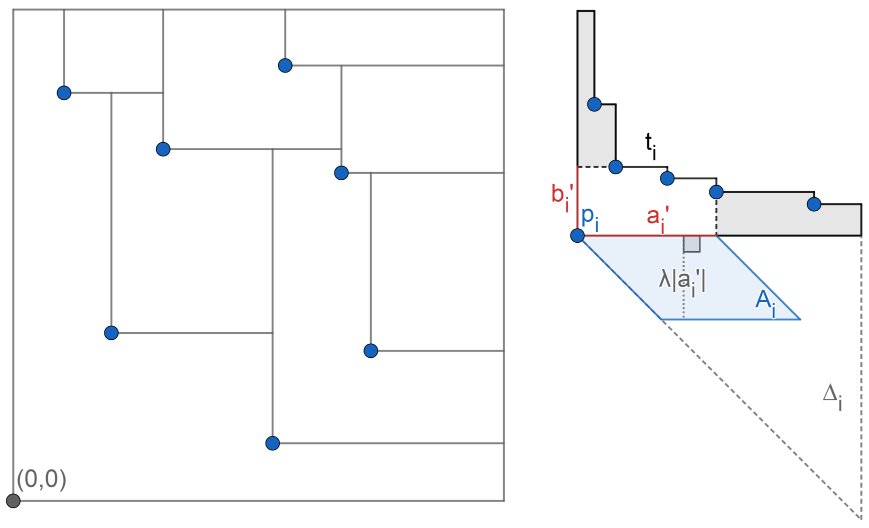


FIGURE 9: Left: the tiling resulting from the tilepacking algorithm on a point set. Right: a wide β-tile t_i, with its right and upper tip in grey. Triangle Δ_i is marked with dashed lines, a'_i and b'_i are in red, and parallelogram A_i is in blue.

as the area of the largest anchored rectangle inside t'_i is less than $\frac{1}{\beta-2-2\alpha}\,\text{area}(t'_i)$, we get from Lemma 3.1 from [6] that $\text{area}(t'_i) < \frac{\beta-2-2\alpha}{e^{\beta-3-2\alpha}}|a'_i||b'_i|$ (this lemma says if $\beta \le 1$ and t is a *staircase polygon*- a polygon with axis-aligned sides, with one horizontal side on the bottom, one vertical side on the left, and a decreasing curve from the top to the right - and t has height h and width w, and every axis-aligned rectangle contained in t has area less than $\frac{\text{area}(t)}{\beta}$, then $\text{area}(t) < \frac{\beta}{e^{\beta-1}}hw$).
Since $|b'_i| \le |a'_i|$, $\text{area}(A_i) = \lambda|a'_i|^2$ and $\text{area}(t'_i) > \frac{\beta-2-2\alpha}{\beta}\,\text{area}(t_i)$, we get

$$\begin{aligned}\text{area}(t_i) &< \frac{\beta}{\beta-2-2\alpha}\,\text{area}(t'_i) < \frac{\beta}{\beta-2-2\alpha}\,\frac{\beta-2-2\alpha}{e^{\beta-3-2\alpha}}|a'_i||b'_i| \\ &\le \frac{\beta}{e^{\beta-3-2\alpha}}|a'_i|^2 = \frac{\beta}{\lambda e^{\beta-3-2\alpha}}\,\text{area}(A_i)\end{aligned}$$

This proves the lemma. □

Next, we prove an upper bound on the total area of all parallelograms A_i of wide β-tiles. To do so, we define a directed graph G, where the nodes correspond to the parallelograms of wide β-tiles. If a parallelogram A_i intersects some other parallelograms, then A_i gets exactly one outgoing edge to the node of the parallelogram A_j that intersects A_i and of which the corresponding line segment a'_j is the highest below a'_i. We say the parallelograms that do not have an outgoing edge are *at level 1*, the parallelograms with an outgoing edge to a level 1 parallelogram, are *at level 2*, and so on. Clearly this is an acyclic graph. Then we use a charging scheme for the parallelograms, where the area of each parallelogram A_i at level 2 and higher is charged to the unique parallelogram at level 1 that it has a directed path to. First, we derive an upper bound on the total area of the level 1 parallelograms:

Lemma 5.7. *The total area of all parallelograms at level 1 is at most* $\frac{2(1+\alpha)^2+\lambda(2+\alpha)}{2(1+\alpha)^2}$.

Proof. The largest rectangle anchored at p_i with a'_i as its bottom edge has area at most $\frac{1}{\beta}\,\text{area}(t_i)$, but has a larger height than the right tip, which has area at least $\frac{\alpha}{\beta}\,\text{area}(t_i)$ (see Figure 9). Therefore $|a'_i| < \alpha(|a_i| - |a'_i|)$, hence $(1+\alpha)|a'_i| < |a_i|$. Note that, as $\lambda < \alpha$, and since the bottom right corner of A_i is exactly $\lambda|a_i'|$ (which is less than $\alpha|a_i'|$) to the right of line segment a_i', this implies that A_i is completely inside Δ_i. Therefore, no parallelogram crosses the right boundary of U. Furthermore, it is very clear that no parallelogram crosses the top or left boundary of U or the line $y = -x$, as the parallelograms have an angle of 45°. Furthermore, since $|a_i| \leq 1$, we have $|a_i'| < \frac{1}{1+\alpha}$, so the height of all parallelograms is less than $\frac{\lambda}{1+\alpha}$. Therefore all parallelograms are above the line $y = -\frac{\lambda}{1+\alpha}$. Finally, note that the bottom right corner of A_i is $\lambda|a_i'|$ below and $\lambda|a_i'|$ to the right of the right end of a_i'. Also, the right end of a_i' is at least $\alpha|a_i'|$ to the left of the right end of a_i. Therefore, the line through the bottom right corner of A_i and the right end on a_i has a slope of at most $\frac{\lambda}{\alpha-\lambda}$ (see left half of Figure 10). As the right end of a_i is inside U, the parallelograms must therefore lie above the line $y = \frac{\lambda}{\alpha-\lambda}(x-1)$. Hence all parallelograms lie in the hexagon bounded by the six lines and edges described before. This hexagon has as corners the corners of U and the points $(\frac{\lambda}{1+\alpha}, -\frac{\lambda}{1+\alpha})$ and $(\frac{1+\lambda}{1+\alpha}, -\frac{\lambda}{1+\alpha})$ (see right half of Figure 10). This hexagon consists of a square and a trapezoid with height $\frac{\lambda}{1+\alpha}$ and base lengths 1 and $\frac{1}{1+\alpha}$, therefore it has an area of $1 + \frac{1}{2}\frac{\lambda}{1+\alpha}\left(1 + \frac{1}{1+\alpha}\right) = \frac{2(1+\alpha)^2+\lambda(2+\alpha)}{2(1+\alpha)^2}$. If two parallelograms at level 1 overlap, then the one with the highest $y-$coordinate of a_i' must have an outgoing edge in G, which is not possible. Hence no two level 1 parallelograms overlap, and their total area is bounded by $\frac{2(1+\alpha)^2+\lambda(2+\alpha)}{2(1+\alpha)^2}$. □

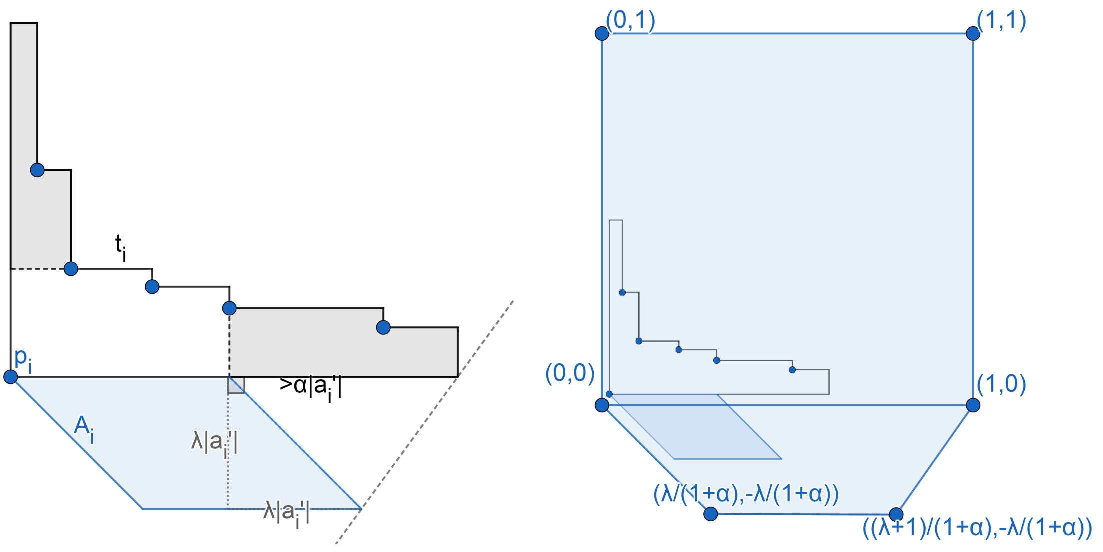


FIGURE 10: Left: the slope of the line through the bottom right corner of A_i and the right end of a_i is at most $\frac{\lambda}{\alpha-\lambda}$. Right: all parallelograms lie inside the blue hexagon.

Now we derive an upper bound on the area of the parallelograms of level 2 and higher. We do this by deriving an upper bound on the area that is charged to the level 1 parallelograms (this is similar to Lemma 3.6 in [6]).

Lemma 5.8. *For every parallelogram A_j at level 1, the total area of all parallelograms A_i, $i \neq j$, with a directed path in G to A_j is at most* $\frac{1}{2(\alpha-\lambda)}$ area(A_j)

Proof. Let l_i be the line through line segment a_i parallel to a_i. Now we consider a parallelogram A_i that intersects a parallelogram A_j, where a_i' is above a_j', and we look at the intersection of Δ_i with l_j (see Figure 11). First, from Lemma 3.3 of [6], we know that there are no points of S in the interior of Δ_i for any point p_i. If the intersection of A_i with l_j contains any point of U left of a'_j, then p_j is inside A_i, hence inside Δ_i, and if it contains any point to the right of a'_j, then the top left corner of the right tip of t_j, which is a point of S, would be inside Δ_i. From this it follows that if a parallelogram A_i intersects a parallelogram A_j, and a'_i is above a'_j, then the intersection of Δ_i with l_j must be a subset of a'_j. Now, we define for a parallelogram A_j, the triangle B_j as the triangle

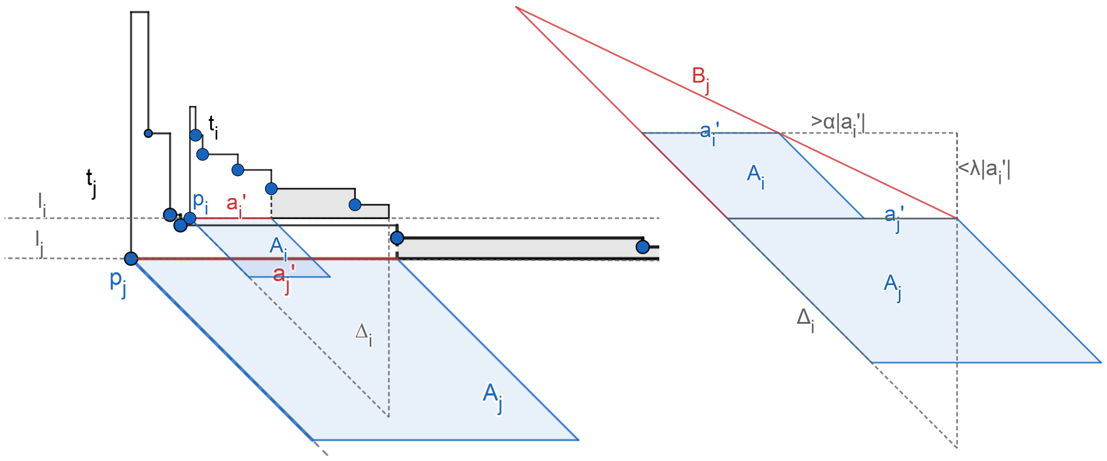


FIGURE 11: Left:parallelograms A_i and A_j intersect, and a'_i is above a'_j. Therefore the intersection of Δ_i and l_j is a subset of a'_j. Right: parallelograms A_i and A_j intersect, and a'_i is above a'_j. Therefore parallelogram A_i fits inside triangle B_j.

that is bounded by a'_j, by the line with slope -1 through p_j and by the line with slope $-\frac{\lambda}{\alpha}$ through the right end of a'_j. See right half of Figure 11.

Consider a parallelogram A_i that intersects parallelogram A_j, and let a'_i be above a'_j. Since we now know that the intersection of Δ_i with l_j is a subset of a'_j, we find that parallelogram A_i must be completely to the right of the left edge of B_j. Furthermore, we know that $(1+\alpha)|a'_i| < |a_i|$, therefore the right edge of Δ_i (and also the right end of a'_j) is at least $\alpha|a'_i|$ to the right of a'_i w.r.t. its $x-$coordinate (see right half of Figure 11). Since we also know that a'_i is at most $\lambda|a'_i|$ (the height of A_i) above a'_j, we find that the right end of a'_i is below the right edge of B_j, as the slope of the line through the right end of a'_i and the right end of a'_j is more than $-\frac{\lambda}{\alpha}$. We conclude A_i is below the right edge of B_j and since A_i is also to the right of the left edge of B_j, we see A_i is completely inside $B_j \cup A_j$ (we refer to this as (i)).

Now let Ξ_i be the strip that is bounded by the two lines tangent to the left and right edge of A_i (see Figure 12). With an induction proof, we see that the strips Ξ_i of the parallelograms A_i at the same level, if the A_i have a directed path to the same parallelogram A_j, are interior-disjoint (we refer to this statement as (ii)).

As induction basis, we show by contradiction that Ξ_{i_1} and Ξ_{i_2} are interior-disjoint if A_{i_1} and A_{i_2} are both at level 2 and have an outgoing edge to the same parallelogram A_j in G. Suppose there are two such parallelograms. If A_{i_1}, A_{i_2} do not intersect, then one parallelogram must be completely above the other, and w.l.o.g. A_{i_1} is above A_{i_2}. But then A_{i_1} cannot intersect A_j, as its bottom edge is above the top edge of A_{i_2} (that is a'_{i_2}), which is above the top edge of A_j (that is a'_j) by definition. So A_{i_1} does not have an outgoing edge to A_j. If, on the other hand, A_{i_1} and A_{i_2} intersect (see Figure 12), then w.l.o.g. a_{i_1} has a higher y-coordinate than a_{i_2}. But then A_{i_1} cannot have an outgoing edge to A_j in G, since a'_{i_2} is above a'_j, so a'_j is not the highest among the parallelograms that intersect A_{i_1}, and this proves the statement.

Now suppose as induction hypothesis that the strips Ξ_i of the parallelograms A_i at level k are all interior-disjoint. From (i) we can see that $\Xi_i \subseteq \Xi_j$ if there is an edge from A_i to A_j in G. The Ξ_i of the parallelograms at level $k+1$ have an outgoing edge to a level k parallelogram, and we can repeat the argument of the induction basis to see that all Ξ_i of A_i, going to the same parallelogram A_{i_0} that is at level k, are interior-disjoint. Combining this with the induction hypothesis, we find that the Ξ_i of the A_i at level $k+1$ are interior-disjoint, and this completes the induction proof.

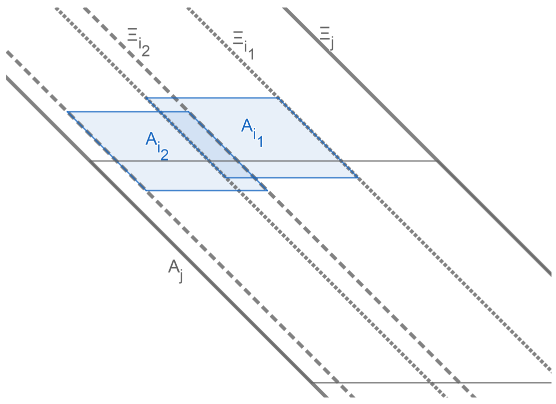


FIGURE 12: The strips Ξ_{i_1}, Ξ_{i_2} and Ξ_j. If the strips Ξ_{i_1} and Ξ_{i_2} overlap, parallelograms A_{i_1} and A_{i_2} overlap, and if a_{i_1} is above a_{i_2}, there cannot be an edge from A_{i_1} to A_j.

Now, for a parallelogram A_j at level 1, we apply a translation on all parallelograms A_i with a directed path to A_j in G, and we claim that after the transformation all parallelograms are inside B_j. We prove this by induction on the levels, using the two statements (i) and (ii). The translation is as follows: first translate all parallelograms at level 2, then those at level 3, 4, etc. When level k is translated, translate all parallelograms one by one, parallel to the line $y = -x$ upwards, and translate each parallelogram exactly far enough so that they do not overlap with any parallelogram at a lower level. By (ii) we know they also do not intersect with their own level, so no two parallelograms overlap at the end of the translation.

For the induction basis, we know that no two parallelograms at level 2 can overlap after the translation. Also, in exactly the the same way as (i), we find that after the translation, a parallelogram A_i is inside B_j (since (i) also holds for a parallelogram A_i that intersects A_j with the intersection area approaching 0). So the parallelograms at level 2 fit inside B_j after translation. Now, suppose as induction hypothesis that all translated parallelograms fit inside B_j after level k is translated. Then, by (ii) we know all Ξ_i of level $k+1$ are interior-disjoint. Also, any parallelogram A_i at level $k+1$ has an edge to a level k parallelogram, say to A_{i_0}. Similar to (i), we find that after the translation, A_i fits inside the translated version of B_{i_0} (if B_{i_0} was translated along with A_{i_0}). Clearly this triangle fits inside B_j, as its bottom edge is inside B_j and its shape is similar to B_j.

Now, from the slopes of the edges of B_j, we find that its height with base a'_j is $\frac{\lambda}{\alpha-\lambda}|a'_j|$, hence its area is $\frac{1}{2}\frac{\lambda}{\alpha-\lambda}|a'_j|^2 = \frac{1}{2(\alpha-\lambda)}\,\text{area}(A_j)$, so the area of all parallelograms with a directed path to A_j is bounded by $\frac{1}{\alpha-\lambda}\,\text{area}(A_j)$ □

Now we can combine Lemma's 5.6, 5.7 and 5.8, to get an upper bound on the area of wide β-tiles. Since an upper bound for tall β-tiles can be derived analogously, we can multiply by 2 to get an upper bound on the area of all β-tiles.

$$\begin{aligned}
\sum_{t_i \text{ is a wide } \beta\text{-tile}} \text{area}(t_i) &\overset{5.6}{<} \sum_{A_i \text{ for } t_i \text{ a wide } \beta\text{-tile}} \frac{\beta}{\lambda e^{\beta-3-2\alpha}} \text{area}(A_i) \\
&\overset{5.8}{\leq} \sum_{A_i \text{ for } t_i \text{ a wide } \beta\text{-tile at level } 1} \frac{\beta}{\lambda e^{\beta-3-2\alpha}} \left(1 + \frac{1}{2(\alpha-\lambda)}\right) \text{area}(A_i) \\
&\overset{5.7}{\leq} \frac{\beta}{\lambda e^{\beta-3-2\alpha}} \frac{1+2\alpha-2\lambda}{2\alpha-2\lambda} \frac{2(1+\alpha)^2+\lambda(2+\alpha)}{2(1+\alpha)^2} \\
\sum_{t_i \text{ is a } \beta\text{-tile}} \text{area}(t_i) &< 2\frac{\beta}{\lambda e^{\beta-3-2\alpha}} \frac{1+2\alpha-2\lambda}{2\alpha-2\lambda} \frac{2(1+\alpha)^2+\lambda(2+\alpha)}{2(1+\alpha)^2} \qquad (6)
\end{aligned}$$

Finally, using the integral equation from chapter 3.3 from [6] (replacing the upper bound on the area of β-tiles $F(\beta,\lambda) = \frac{(3-3\lambda+\lambda^2)(8+3\lambda-\lambda^2)e^5}{2\lambda(1-\lambda)(2-\lambda)} \frac{\beta}{e^\beta}$ by $2\frac{e^{3+2\alpha}}{\lambda} \frac{1+2\alpha-2\lambda}{2\alpha-2\lambda} \frac{2(1+\alpha)^2+\lambda(2+\alpha)}{2(1+\alpha)^2} \frac{\beta}{e^\beta}$, and replacing the condition $\beta_0 \geq 5$ by $\beta_0 \geq 3+2\alpha$), we get for the worst case area ρ, and for some $\beta_0 \geq 3+2\alpha$:

$$\rho \geq \frac{1}{\beta_0} - \frac{(1+2\alpha-2\lambda)(2(1+\alpha)^2+\lambda(2+\alpha))}{(1+\alpha)^2\lambda(2\alpha-2\lambda)} e^{3+2\alpha} \int_{\beta_0}^{\infty} \frac{1}{te^t} dt$$

Taking the values $\alpha = 0.76981$, $\beta_0 = 9.39405$ and $\lambda = 0.44588$ yields $\rho \geq 0.09612$. Note that this also implies that the worst case approximation ratio is at least 0.09612. This yields the following theorem:

Theorem 5.9. *The greedypacking algorithm with ordering according to $x+y$ from high to low yields an anchored rectangle packing of area at least 0.09612 for any finite point set including the origin.*

6 Conclusions

In this thesis orderings for the greedypacking algorithm that perform well were researched. First, there are three assumptions that can be made for any ordering:

- The points that are not dominated by any other point come first in the ordering, and the order among these points can be arbitrary.
- The origin comes last in the ordering
- The ordering satisfies the *dominance property:* for all $p, q \in S$ where p dominates q, p comes before q in π'.

A number of different orderings were compared to an algorithm that finds the optimal anchored rectangle packing. From empirical experiments with orderings according to a function $g : U \to \mathbb{R}$, it seemed that functions that increase with x that have concave level curves have the highest probability to yield optimal orderings. However, no such ordering always yields the optimum. For this reason, three variations of a new algorithm were proposed, of which the ordering depends on the relative positions of the points.

A formula for the average case performance in terms of area of resulting ARP was created for sets of size 4, and, and the average case performance was analyzed empirically for larger point sets. For the larger sets, the ordering according to $\|(x,y)\|_1 = \frac{x+y}{2}$ was either the best or among the best performing orderings, depending on the size of the point sets. All orderings had a high approximation ratio on average: about 0.98 to 0.99.

Finally, some upper bounds and a lower bound on the worst case approximation ratios of the orderings were established. These bounds are summarized in the Table 1.

TABLE 1: Upper and lower bounds on the worst case absolute performance and on the worst case approximation ratio of different orderings.

Ordering	Absolute		Approximation ratio	
	L. bound	U. bound	L. bound	U. bound
Symmetric $g : U \to \mathbb{R}$ that satisfies dominance property	0	$\frac{1}{2}$	0	$\frac{3}{4}$
$\min(x, y)$ Area ordering	0	$\frac{1}{2}$	0	$\frac{1}{2}$
Euclidean ordering Combined ordering	0	$\frac{1}{2}$	0	$1 - \frac{\sqrt{2}}{4} \approx 0.646$
$\|(x, y)\|_1$	0.09612	$\frac{1}{2}$	0.09612	$\frac{3}{4}$

7 Discussion and recommendations

The best results were found with an ordering that was known already: ordering according to $\|(x, y)\|_1$. This may be the best of all functions $g : U \to \mathbb{R}$ for which the ordering satisfies the dominance property. In finding even better orderings, these can probably be found in orderings that incorporate relative point locations. Using a different function than $\|p\|_2$ might for example improve the combined ordering. Furthermore, the time complexity of $\Theta(N_S^3)$ of the three variants of relative position-based algorithms presented here can probably be improved, as in the current implementation the same distances and areas are calculated many times.

For the simulations on the performance, only uniform point distributions were used. In the comparison to optimal orderings, we saw that the performance of all algorithms was dependent on the distribution, and the results found for the average performance is very likely to be different for different distributions (although the relative performance between the algorithms might stay quite the same). Furthermore, note that the comparison with the optimum was only done for small point sets due to large computation times. The results there might be slightly different for large sets.

About the upper bounds on the worst case performance: it is likely that the upper bound of $\frac{3}{4}$ for the worst-case approximation ratio of the ordering that satisfies the dominance property according to an arbitrary function $g : U \to \mathbb{R}$, can be sharpened to $\frac{2}{3}$ by a small adaptation of the proof. Also, it seems very likely that the bound of $\frac{1}{2}$ for $\min(x, y)$ is tight, and that the worst-case approximation ratio for all these functions g is between $\frac{1}{2}$ and $\frac{2}{3}$. Finally, no cases were found where any of the used algorithms yielded an area less than $\frac{1}{2}$, so also the bound for the absolute performance may be tight.

The main idea for increasing the lower bound on the performance of the greedypacking algorithm with $\|(x, y)\|_1$ was to apply the observation that triangle Δ_i does not contain points of S to a larger part of Δ_i in a better way, and use the β-tiles with smaller β more. Still, a large part of Δ_i is not used, and the fact that its dual Γ_i (the isosceles right triangle on the left edge of t_i) is empty of points of S is not used either, neither are the β-tiles with $\beta < 9.39$. Using these more could increase the lower bound of 0.09612 by a large factor.

References

[1] Antonios Antoniadis, Andrés Cristi, Ruben Hoeksma, Peter Kling, Felix Biermeier, Christoph Damerius, Dominik Kaaser, and Lukas Nölke. On the complexity of anchored rectangle packing. In *Leibniz International Proceedings in Informatics, LIPIcs*, volume 144. Schloss Dagstuhl- Leibniz-Zentrum fur Informatik GmbH, Dagstuhl Publishing, sep 2019. doi:10.4230/LIPIcs.ESA.2019.8.

[2] Kevin Balas and Csaba D. Tóth. On the number of anchored rectangle packings for a planar point set. *Theoretical Computer Science*, 654:143–154, nov 2016. doi:10.1016/j.tcs.2016.03.007.

[3] Kevin Balas, Adrian Dumitrescu, and Csaba D. Tóth. Anchored rectangle and square packings. *Discrete Optimization*, 26:131–162, nov 2017. ISSN 15725286. doi:10.1016/j.disopt.2017.08.003.

[4] Therese Biedl, Ahmad Biniaz, Anil Maheshwari, and Saeed Mehrabi. Packing boundary-anchored rectangles and squares. *Computational Geometry: Theory and Applications*, 88, jun 2020. doi:10.1016/j.comgeo.2020.101610.

[5] Tobias Christ, Andrea Francke, Heidi Gebauer, Jiří Matoušek, and Takeaki Uno. A Doubly Exponentially Crumbled Cake. *Electronic Notes in Discrete Mathematics*, 38: 265–271, dec 2011. doi:10.1016/j.endm.2011.09.044.

[6] Adrian Dumitrescu and Csaba D. Tóth. Packing anchored rectangles. *Combinatorica*, 35(1):39–61, feb 2015. doi:10.1007/s00493-015-3006-1.

[7] Jonathan Gadea Harder. Anchored Rectangle Cover. *SSRN Electronic Journal*, dec 2019. doi:10.2139/ssrn.3463959.

[8] IBM. Ponder this. `https://www.research.ibm.com/haifa/ponderthis/challenges/June2004.html`, 2004.

[9] Michele Muller-Itten. Packing Rectangles: A Cake Sharing Puzzle. *SSRN Electronic Journal*, aug 2019. doi:10.2139/ssrn.3426472.

[10] W. Tutte. Recent progress in combinatorics: Proceedings of the 3rd waterloo conference on combinatorics. page 345, 1969.

[11] Peter Winkler. *Mathematical mind-benders*. CRC Press, 2007.

A Source code and data sets for Python

The used Python programs, the data sets and the results of the simulations of the performance analysis can be found with the following link: `https://git.snt.utwente.nl/s1845004/bachelor-thesis`

B More results of simulation dppacking

This appendix contains the results of the experiments that approximate $F_{f_X,N}(p)$. For all experiments, U was divided into 100×100 small squares. Furthermore, there is a table of the simple counting experiment.

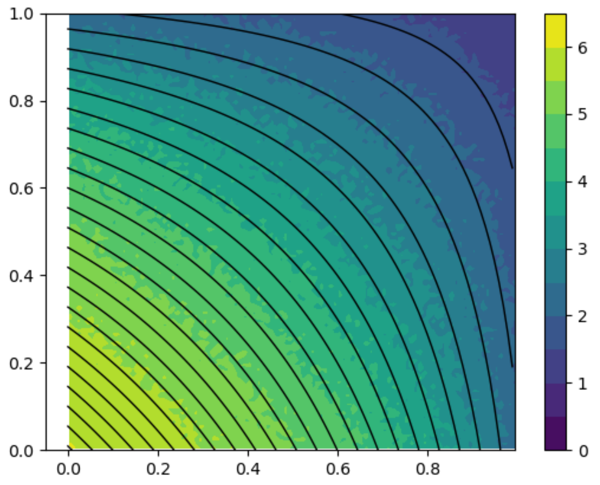


FIGURE 13: Contour plot for $X \sim Uniform(0,1)$, for $N = 7$ and 150000 generated point sets. Some level curves of $\sqrt{(1.1-x)(1.1-y)}$ are drawn over the plot (label 0 means no data).

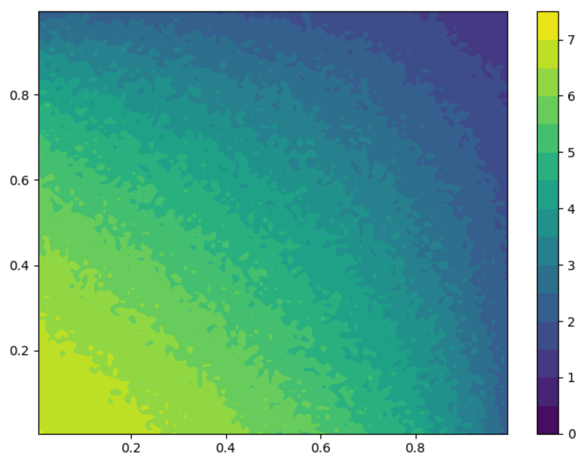


FIGURE 14: Contour plot for $X \sim Uniform(0,1)$, for $N = 8$ and 100000 generated point sets (label 0 means no data).

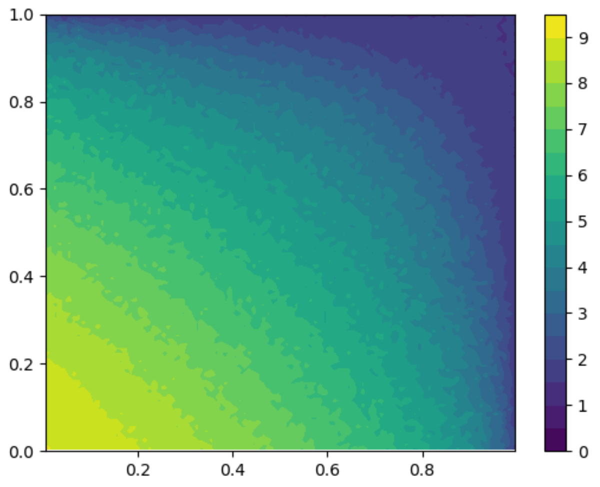


FIGURE 15: Contour plot for $X \sim Uniform(0,1)$, for $N = 10$ and 150000 generated point sets, but where only sets are considered with exactly 2 points that are not dominated by any other point (label 0 means no data).

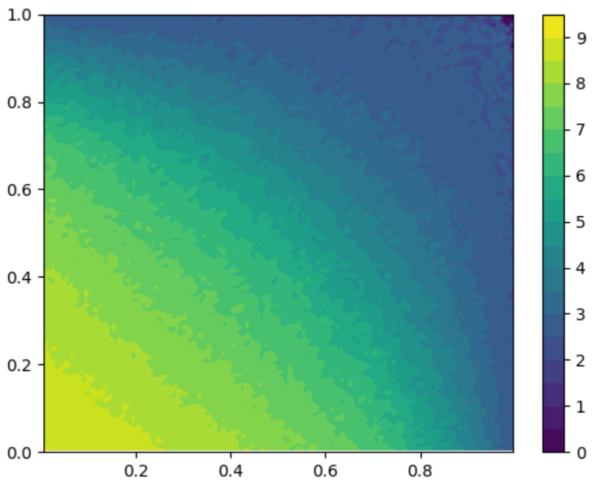


FIGURE 16: Contour plot for $X \sim Uniform(0,1)$, for $N = 10$ and 150000 generated point sets, but where only sets are considered with exactly 4 points that are not dominated by any other point (label 0 means no data).

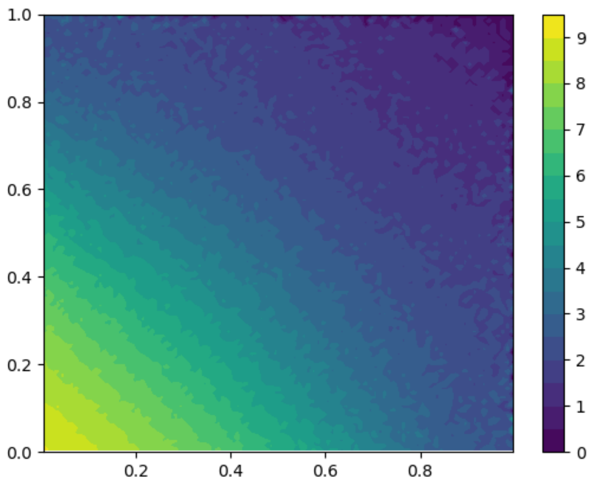


FIGURE 17: Contour plot for X distributed according to a triangular distribution with range $[0,1]$ and mode 0, with division of U into $100{\times}100$ small squares, for $N = 10$ and 150000 generated point sets. (label 0 means no data)

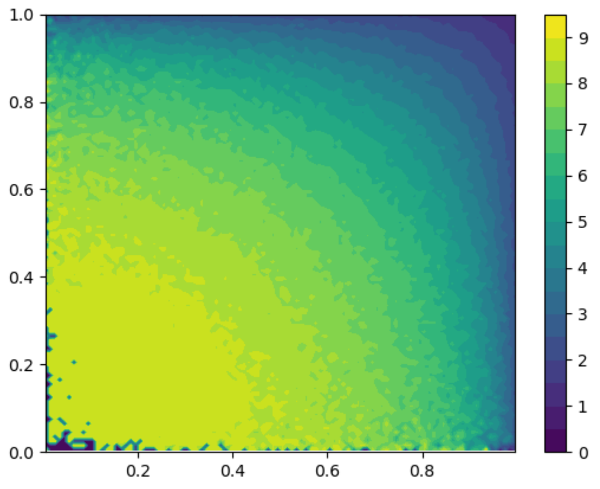


FIGURE 18: Contour plot for X distributed according to a triangular distribution with range $[0, 1]$ and mode 1, for $N = 10$ and 150000 generated point sets (label 0 means no data).

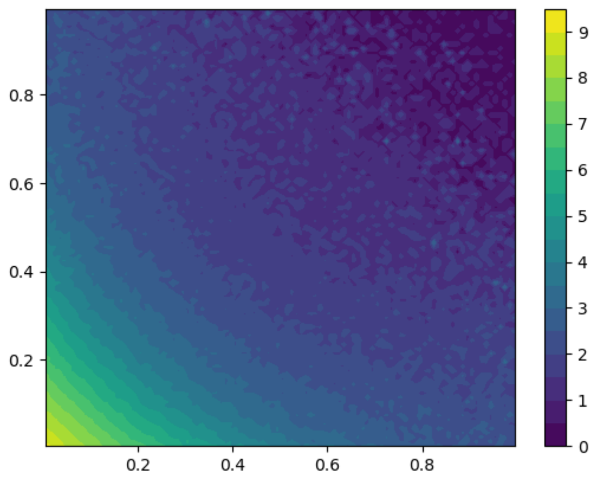


FIGURE 19: Contour plot for X distributed according to conditional distribution of en exponential random variable with mean $1/5$, with the condition $X \in [0, 1]$, for $N = 10$ and 150000 generated point sets (label 0 means no data).

TABLE 2: Number of optimal orderings for ordering by some functions of x and y (from high to low) over 200000 randomly generated sets S of size 10. The x and y coordinates were independent and distributed like a random variable X, for three distributions: uniform with range $[0, 1]$, triangular with range $[0, 1]$ and mode 0, and exponential with mean $\frac{1}{5}$, conditional to $0 < X < 1$. Some norms of $(1-x, 1-y)$ are left out because sorting by minus the $1, -\infty, \infty$-norms of $(1-x, 1-y)$ is equivalent to sorting by resp. $1, \infty, -\infty$- norms of (x, y).

Ordered by:	Number of optimal orderings		
	Uniform	Triangular	Exponential
Random	23290	20028	17992
$\|(x,y)\|_{-\infty} = \min(x,y)$	20398	16133	14211
$\|(x,y)\|_{-2} = \sqrt{\frac{2}{\frac{1}{x^2}+\frac{1}{y^2}}}$	23157	19053	16854
$\|(x,y)\|_{-1} = \frac{2}{\frac{1}{x}+\frac{1}{y}}$	25294	21935	20055
$\|(x,y)\|_0 = \sqrt{xy}$	30340	29575	30746
$\|(x,y)\|_1 = \frac{x+y}{2}$	39167	41714	44152
$\|(x,y)\|_2 = \sqrt{\frac{x^2+y^2}{2}}$	41679	43787	44940
$\|(x,y)\|_{\infty} = \max(x,y)$	37822	39803	41533
x	15530	11406	8980
y	15470	11492	9112
$-\|(1-x, 1-y)\|_{-2}$	41130	43643	44618
$-\|(1-x, 1-y)\|_{-1}$	41733	43717	44516
$-\|(1-x, 1-y)\|_{0}$	41926	43361	44444
$-\|(1-x, 1-y)\|_{2}$	33531	38362	43465
$\frac{x+y-\lvert x-y\rvert}{1-\lvert x-y\rvert}$	34631	26797	20467

C Tables with results performance analysis

This appendix contains the results for the average and worst case performance analysis.

Table 3: Average case performance of different orderings. In the top rows, the orderings are sorted by average performance at $N_S = 10$, and in the bottom rows, for average performance at $N_S = 50$. A uniform distribution of points was used (except for the origin).

Ordering	$N_S = 10$, for 10^5 sets				$N_S = 25$, for 10^5 sets	
	Absolute		As part of optimum		Absolute	
	Average	σ	Average	σ	Average	σ
$-\|(1-x,1-y)\|_0$	0.83459	0.04780	0.98830	0.01556	0.84138	0.03212
$\|(x,y)\|_1$	0.83424	0.04782	0.98797	0.01572	0.84176	0.03188
$\|(x,y)\|_2$	0.83417	0.04781	0.98787	0.01616	0.83931	0.03220
$-\|(1-x,1-y)\|_{-1}$	0.83406	0.04793	0.98771	0.01624	0.83746	0.03287
$-\|(1-x,1-y)\|_{-2}$	0.83348	0.04803	0.98703	0.01694	0.83397	0.03344
$-\|(1-x,1-y)\|_2$	0.83300	0.04807	0.98649	0.01716	0.83896	0.03230
Combined ordering	0.83208	0.04821	0.98534	0.01891	0.83678	0.03231
$\|(x,y)\|_\infty$	0.83124	0.04803	0.98433	0.01951	0.82506	0.03409
Euclidean ordering	0.83014	0.04855	0.98304	0.02142	0.83331	0.03276
Area ordering	0.82449	0.04977	0.97660	0.02770	0.82433	0.03465

Ordering	$N_S = 50$, for 25000 sets		$N_S = 100$, for 10^4 sets	
	Absolute		Absolute	
	Average	σ	Average	σ
$\|(x,y)\|_1$	0.85249	0.02262	0.86239	0.01689
$-\|(1-x,1-y)\|_0$	0.85093	0.02292	0.86022	0.01728
$-\|(1-x,1-y)\|_2$	0.84877	0.02296	0.85840	0.01717
$\|(x,y)\|_2$	0.84723	0.02318	0.85524	0.01759
Combined ordering	0.84622	0.02306	0.85539	0.01768
Euclidean ordering	0.84326	0.02337	0.85353	0.01785
$-\|(1-x,1-y)\|_{-1}$	0.84290	0.02409	0.84901	0.01825
$-\|(1-x,1-y)\|_{-2}$	0.83588	0.02497	0.83895	0.01903
Area ordering	0.82897	0.02594	0.83287	0.02017
$\|(x,y)\|_\infty$	0.81916	0.02619	0.81343	0.02042

TABLE 4: Worst case performance in the simulation of different orderings, both absolute and relative to the optimum.

Ordering	$N_S = 7$, for $2 \cdot 10^5$ sets		$N_S = 10$, for 10^5 sets		$N_S = 25$, for 10^5 sets
	Absolute	Relative	Absolute	Relative	Absolute
$-\|(1-x, 1-y)\|_0$	0.63721	0.85663	0.64242	0.86083	0.68262
$\|(x, y)\|_1$	0.63721	0.85507	0.63997	0.86083	0.68714
$\|(x, y)\|_2$	0.63721	0.85663	0.63997	0.84601	0.67958
$-\|(1-x, 1-y)\|_{-1}$	0.63721	0.85663	0.64242	0.85937	0.68439
$-\|(1-x, 1-y)\|_{-2}$	0.63721	0.85663	0.64242	0.85201	0.68621
$-\|(1-x, 1-y)\|_2$	0.63721	0.84864	0.63326	0.86353	0.67222
Combined ordering	0.63371	0.81753	0.64242	0.83001	0.67574
$\|(x, y)\|_\infty$	0.63721	0.84280	0.65355	0.83276	0.68421
Euclidean ordering	0.63371	0.80371	0.64242	0.81579	0.67270
Area ordering	0.63371	0.73681	0.61587	0.75561	0.66794