Rectangle Packing

Investigating placement of rectangles at the edge of a container
relative to the rectangle size

Floris de Bruin
5772583

Bachelor thesis
Credits: 18 EC

Bachelor Opleiding Kunstmatige Intelligentie

University of Amsterdam
Faculty of Science
Science Park 904
1098 XH Amsterdam

Supervisor
Dhr. drs. D. van den Berg

Faculty of Science
University of Amsterdam
Science Park 904
1098 XH Amsterdam

June 28th, 2013

Abstract

Rectangle packing problems are NP-complete problems, this property makes finding solutions for a packing problem time consuming. Investigating rectangle packing problems could lead to better understanding of the problems, better algorithms and new heuristic. This thesis investigates the assumption that large rectangles are placed at the side of a container.

Existing solved almost square rectangle packing problems where the rectangle sets can be packed perfectly, and creating new closely related rectangle packing problems generates the investigated data.

The results presented indicate a strong correlation for the size of a rectangle area an placement on the side. For the length of a side of the rectangle there is a weaker correlation. The probability of placement on the side for a rectangle with a relative large area in almost square rectangle packing is between 99 and 100 percent. For the closely related rectangle sets this possibility is around 90 percent.

Contents

Acknowledgements

Foremost, I would like to express my sincerest gratitude to my supervisor Dhr. drs. D. (Daan) van den Berg for his support of my thesis and research, for his enthusiasm, motivation and time. Because of his guidance I was able to overcome the many obstacles this thesis research created. I thank SURFsara (www.surfsara.nl) for the support in using the LISA Compute Cluster and guidance in submitting the batch tasks. Finally, I would like to thank everybody that was involved during the accompanying BSc. graduate project course: dhr. dr. B. Bredeweg, dhr. dr. A. Visser, dhr. dr. M.W. van Someren and all students present at my project presentations.

1 Introduction

When presented with a set of rectangles and a container, is there a configuration where all rectangles can be placed within the container without overlapping each other or the edge of the container? This problem is know as a *rectangle packing problem*. An example of a rectangle packing problem can be seen in figure 1.

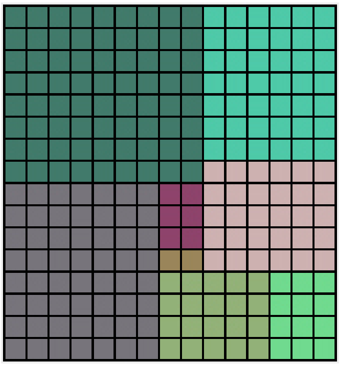


Figure 1: Rectangle Packing: 8 Almost Square Rectangles inside a 15×16 container

Rectangle packing problems are a subset of *packing problems* where the goal is to pack a single container or multiple containers as efficient as possible with different shapes and sizes. Packing problems are applicable to multiple real world problems; efficient packing of objects for transport, placing parts of a computer chip as close to each other as possible to reduce wire length, the efficient cutting of fabric to reduce waste in the clothing industry, and reducing the file size of CSS sprites [Perdeck, 2011] for websites are some examples of real life applications of packing problems.

Rectangle packing problems are know as NP-complete problems. The NP-completeness characteristic is proven by Richard Korf [Korf, 2003] by reducing the rectangle packing problems to bin-packing problems. Bin-packing problems are a subset of packing problems where objects of different shapes and sizes are packed into a finite number of containers with a fixed volume. This is done while minimizing the number of used containers. Because of the NP-completeness characteristic solving rectangle packing problems is a time consuming task, even for the most known efficient algorithms, and becomes exponentially harder when the number of rectangles increase.

In this study, multiple different techniques are used to aid the search for solutions. In section 1.1 some of these techniques are mentioned. These techniques are mostly quasi-human based heuristics and reduce the number of tries needed to find a solution for the rectangle packing problem as often demonstrated. There is an assumption, within the family of rectangle packing problems, that large rectangles are persistently placed at the side of a container. However, there is not enough evidence to support this assumption.

The prospect of accelerating rectangle packing solvers justifies the research conducted for this thesis. As mentioned in the previous paragraph there is not enough evidence to solidify the assumption of placement of large rectangles at the edge of a container. The placement of all rectangles in multiple solved rectangle packing problems could be analyzed to possibly create a sound basis for future packing algorithms. The thesis question therefore is as follows; are large rectangles more frequently placed on the side of a container?

To examine the thesis question, a substantial number of rectangle packing problems are needed to create a large enough data set to support statistical plausibility. Thus, a packing solver is created to generate information for statistical analysis (section 2.3), a rectangle set creator generates rectangle sets based on almost square rectangle sets (section 2.1), and a container analyzer evaluates the placement of the rectangles inside the container (section 2.4).

The search space is limited to *perfect* rectangle packing problems, perfect means there is no empty space between rectangles. The opposite of perfect packing is *non-perfect* packing where there is empty space allowed between rectangles. There is also a limit on the rectangle sets. Only almost square rectangle sets up 20 rectangles are considered, and rectangle sets that are closely related tot almost square rectangle set are evaluated with up to size 16. This is to restrain solving time and gives the opportunity to utilize the characteristics of consecutive almost square rectangles, rectangles set with rectangles of sizes $1 \times 2, 2 \times 3, ..., n \times n+1$ where n is the size of the set (figure 1 is an example of almost square packing with eight rectangles). Sets of this kind are often denoted as AS-N, e.g. AS-8 for eight consecutive almost squares. These sets are solvable for certain containers, as given by Simonis and O' Sullivan [Simonis and O'Sullivan, 2011]. The formation of closely related sets is explained in section 2.2 and is done to increase the data set size for the same solvable containers of almost square rectangle packing. Because this thesis only involves placing rectangles in a known container the problems are specified as containment problems not minimal bounding box problems (the difference is explained in section 1.1.4). Non-perfect packing problems are not included due to the fact that solving these problems would need a different approach in the development of the packing algorithm.

This thesis is organized as follows. First previous literature is recalled to explain the techniques that are used in the rectangle packing solver. Then, the methods for generating results are explained, e.g. the container analyzer and the rectangle set creator. Next, the results are presented in detail. It is explained how the results show a strong correlation between rectangle size and placement on the side. Next, a possible cause is presented for this correlation. Finally, the possible implementation of this results is discussed and possible improvements.

1.1 Previous Work

Rectangle packing problems are a well known set of problems within mathematics and heuristics. In the late eighties experiments started to calculate packing for given containers. The development of the packing solver for this thesis is influenced by earlier work explained in the next sections.

1.1.1 Depth-First

One of the earlier examples of an algorithm to solve the rectangle packing problem was described by Katheryn Dowsland (1987) [Dowsland, 1987]. She described an exact algorithm for finding optimal layouts for identical rectangular boxes on a rectangular pallet. The described algorithm was able to handle cases up to 50 boxes of the same size per layer, but was only efficient in packing 30 boxes or less. By only including same size boxes the problem was significantly simpler to solve than later, more general, algorithms. By describing the search space as an abstract graph it was possible to solve the problem with a depth-first search algorithm. In following years of research, depth-first search was used as an effective method for solving rectangle packing problems.

1.1.2 Branch-and-boud placement

In the article *Branch-and-bound placement for building block layout*, by Onodera et al.

[Onodera et al., 1991], a method is described for placing different sized rectangular blocks in a container. The presented method searches an optimal solution in the whole solution space. There is no differentiation between perfect or non-perfect packing. The branch-and-bound strategy constrains the search for the solution by taking in the space available for placement after a placement of each rectangle. This method was able to handle around six rectangles in a reasonable CPU time when published in 1991.

1.1.3 Less Flexibility First

All continuing research regarding the rectangle packing problem uses the branch-and-bound placement pruning method for the development of new pruning techniques. As research progressed the rectangle packing problem was solved for bigger rectangle sets in bigger solution spaces. A proposed deterministic heuristic method, introduced by Wu et al. [Wu et al., 2002], is Less Flexibility First. It is suggested that the empty space with "less flexibility" should be filled earlier than other empty spaces. This algorithm can produce rectangle packings with densities of around 99% for randomly generated sets of rectangles greater than 45. This is a method for finding the smallest container for a given set of rectangles, a minimal bounding box problem. What a minimal bounding box problem is will be explained in the next section.

1.1.4 Containment and Minimal Bounding Box Problems

In the following years the rectangle packing problem was divided into the containment problem and the minimal bounding box problem. The divide is first described by, and accounted to, Korf [Korf, 2003] and Simonis et al. [Simonis and OSullivan, 2008]. In the containment problem a given set of rectangles is to be packed in a given container, as opposed to the minimal bounding box where the smallest container is found for the given set of rectangles. The containment problem is found to be a sub-problem for the minimal bounding box problem. In the article by Huang and Korf [Huang and Korf, 2012] the most successful methods of solving both rectangle packing problems are evaluated and tested for performance. In the specific example of consecutive almost square rectangles (rectangles of sizes 1×2, 2×3, ..., and $N \times (N+1)$) packing is shown to be solvable for rectangle sets up to size 29.

2 Methods

To create the results for this thesis multiple components were developed. In this section these components are explained in detail.

The solving program and its components were developed with the Java Development Kit 1.6. To utilise the full potential of the LISA Cluster of SURFsara, that performed a large portion of the calculations on the large sets, the solver uses multi-threading for 16 cores of the Intel® Xeon E5-26520 v2 at 2.60 GHz. For solving packing of smaller sets and result analysis, Intel® i7-3610QM with 8 cores where used running at 2.30 GHz. The analysis of the results of the packing were done in MATLAB (version R2013a).

2.1 Almost Square Rectangle Sets and their Containers

Almost square rectangle sets are sets with sizes 1×2 to $n \times (n+1)$. The surface area of almost

Table 1: Almost Square Rectangle Set with container sizes that can be perfectly packed

Size N	Container Size	Empty Space
10	17×26	0.45%
11	22×26	0.00%
12	21×35	0.95%
13	26×35	0.00%
14	32×35	0.00%
	28×40	0.00%
15	34×40	0.00%
16	32×51	0.00%
17	34×57	0.00%
18	30×76	0.00%
19	35×76	0.00%
	38×70	0.00%
20	35×88	0.00%
	44×70	0.00%

square rectangles are also in increasing order. The orientation of these rectangles can be freely chosen when placing these rectangles inside a container. But because the goal of packing is to find an optimal solution, the orientation is limited to regular (0°) or rotated 90° clockwise around the origin.

Almost square rectangle sets are solvable for certain containers. Huang and Korf published the containers given in Table 1 in 2012 [Huang and Korf, 2012]. The sets AS-10 and AS-12 do not appear to fit exactly into a container, there is empty space between the rectangles, and thus have not been part of this research.

2.2 Creation of Closely Related Rectangle Sets

To guarantee a large data set for a a single container from Table 1, a method for creating possible solvable rectangle sets was designed. By preserving the area of each rectangle and varying the whidth and height, new rectangle sets can be created. It appeared to be possible to pack these newly created rectangle sets in the same container as the corrosponding AS-N set.

The creation of the closely related rectangle sets always follow the same rules. To create a new set, one rectangle from the original set is mutated by a factor. This means that one rectangle side is divided by factor x and the other side is multiplied by the same factor x. For this thesis a constant factor of two is chosen, this is because this factor is a divisor in every rectangle in one side or the other.

Consider the rectangle set AS-11 (1×2, 2×3, ..., 11×12), the first mutation by factor two would result in the following set of rectangles;

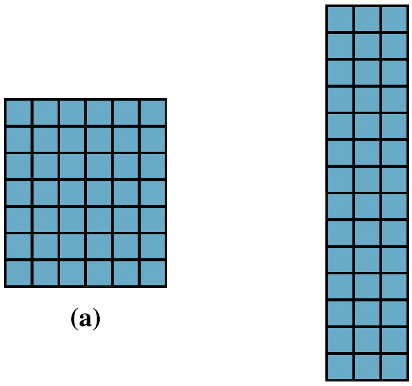
$$\{(1 \times 2), (1 \times 6), (3 \times 4), ..., (11 \times 12)\}$$

The next mutation set would be;

$$\{(1 \times 2), (2 \times 3), (6 \times 2), ..., (11 \times 12)\}$$

In Figure 2 an example is given of a non mutated rectangle of 6×7 (Figure 2a) and its mutated counterpart 3×14 (Figure 2b). Both rectangles have the same area, 42 but have different widths and heights.

This creates a large number of new rectangle sets. The collection of sets that can be created from AS-11 with the mutation factor of two is 2^{10}(=1024). A rectangle either be mutated or not, thus the base number 2 and because the first rectangle is not mutatable the power can be written as $n-1$ where n is equal to the number of rectangles in the set. This creates a general rule of number of mutations as 2^{n-1}.



(a) (b)

Figure 2: Example of a non mutated 6×7 rectangle (2a) and mutated (2b) rectangle. The mutation is done by a factor of 2.

2.2.1 Number of Possible Closely Related Rectangle Sets

Beyond the rectangle sets that are used in this investigation there are even more rectangle sets that can be created and could be used. The total number of possible closely related sets is far greater than the 2^{n-1} mentioned.

The creation of these sets follow a pattern and are related to the number of divisors for each rectangle area. It is possible to find number of divisors for eache rectangle area via a divisor function;

$$\sigma_x(n) = \sum_{d|n} d^x$$

where $d|n$ stands for d divides n.

In this case x is 0 because we only need to know how many divisors there are. For the example of a rectangle with sides 6×7 and area of 42;

$$\begin{aligned}\sigma_0(42) &= \sum_{d|42} d^0 \\ &= 1^0 + 2^0 + 3^0 + 6^0 + \ldots \\ &\quad 7^0 + 14^0 + 21^0 + 42^0 \\ &= 8\end{aligned}$$

Because the total number of divisors gives pairs of sides, 1×42, 2×21 and so on, we need to divide the total number of divisor by two to find the unique possible rectangles. The number of possible mutations for the rectangle 6×7 is eight divided by two, thus 4. This gives the general function for a single rectangle N with sides n and $n+1$;

$$\sigma_0(N(area))/2$$

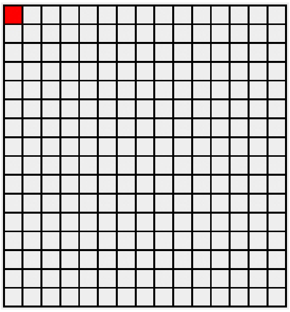
To find the number of all the possible mutations therefore is calculated by multiplying the number of possible mutations for all rectangles N in the set;

$$\prod_{n=1}^{N} \left(\sigma_0\left(N\left(area\right)\right)/2\right) \tag{1}$$

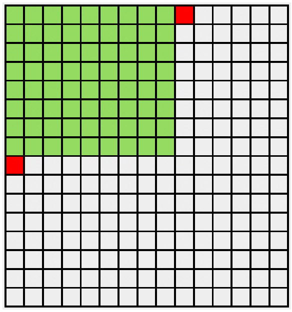
The total number of rectangles that are possible to create and still are closely related to almost square rectangle sets is now defined as mathematical function, Table 2 gives an overview of the sets 11 to 16 where the original AS-N problem has a perfect fit solution. In Section 5.1 it will be discussed how this can be further used in research.

Table 2: Number of possible closely related rectangle sets with all mutation factor according to Equation 1

Size N	Possible Sets
11	995328
13	23887872
14	191102976
15	1911029760
16	9555148800



(a) Begin situation with the insertion point at coordinate $(0,0)$.



(b) Situation after placing 8×9 rectangle with the insertion points at coordinates $(0,8)$ and $(9,0)$.

Figure 3: Example of placing a rectangle of size 8×9 on a field and the insertion points for new rectangles marked in red for AS-8 in a 15×16 container.

2.3 Rectangle Packing Solver

For the rectangle packing solver the container is divided into squares of 1×1. This is to create a method of expressing exact placement of a rectangle in a container. Where an x and y value make up a coordinate. The coordinates that express the location in the container are zero indexed starting at the upper-left corner. Coordinate $(0,0)$ is marked in red in Figure 3a.

2.3.1 Placing Rectangles

A single rectangle of size 8×9 can be placed in 127 different ways in an empty 15×16 container. In a row the non-rotated 8×9 rectangle can be placed in 7 different locations. In a column the same rectangle can be placed in 9 different locations. This means that the non-rotated rectangle has 63 different locations. If we calculate the same values for the rotated rectangle, there are 64 different locations. By adding these values we get the 127 different locations. In general, the rule for possible placements of all rectangles on an empty field is;

$$(W_{\text{container}} - W_{\text{rectangle}}) \\ \times \\ (H_{\text{container}} - H_{\text{rectangle}})$$

for every rectangle and rectangle orientation, where W represents width, and H represents height.

To reduce this number of different locations a rectangle can be placed on an empty field, a static coordinate is chosen, an insertion point. An insertion point is a coordinate where a rectangle can be placed. By choosing insertion points, the number of iterations needed to find a space for a rectangle is reduced. In the empty field example (Figure 3a), the insertion point is coordinate $(0,0)$. After the placement of a rectangle at the insertion point new insertion points are created. These insertion points are next to the edge of the last placed rectangle, on the down-left side and the up-right side, or $(\text{rectangle } x, \text{rectangle } y + \text{rectangle height})$ and $(\text{rectangle } x + \text{rectangle width, rectangle } y)$. In Figure 3b, the insertion points are updated to coordinates: $(0,8)$ and $(9,0)$. After every placement of a rectangle inside the container these points are updated.

For every rectangle there are two orientations, normal and 90° rotated clock-wise. In an AS-N problem this produces $2 \times N$ configurations for the begin insertion point in an empty container. So, for an AS-8 problem the beginning state can lead to 16 new states. To find one solution in the optimal conditions there are only N states necessary. Because all configurations that give a solution are needed for analysis the whole search space needs to be traversed, and thus

$$N! \times 2$$

configurations are traversed in the search space. As the N in an AS-N problem increases the number of possible configurations increases exponentially.

2.3.2 Pruning

To reduce the number of nodes in the state-space that need to be traversed, pruning techniques need to be implemented. The simplest implementation of a pruning technique is to stop going deeper into the search tree when a situation arises where it is not possible to place any rectangle on the insertion points left in the container.

The second method of pruning that is used is the Less Flexibility First algorithm [Wu et al., 2002]. An insertion point is an indicator of a packing space. By finding the width and the height of the space defined by the insertion point, the least flexible space is found. In this thesis the least flexible space is chosen based on the smallest side of the packing space. It could be argued that choosing the smallest side as the value that defines the packing space is not accurate enough, and what should be chosen a the defining value is the area. When testing both methods of calculating the less flexible value in the research stage, the results showed that in some situations the area was a better indicator and in other cases the smallest side was the better criterion. The area of the least flexible space is a more difficult calculation and takes more time, and because the calculation is done repeatedly the less complicated method was used.

In Figure 3b there are two insertion points. Coordinate $(9,0)$ has a smallest side value of 6, equal to the width of the free space. Coordinate $(0,8)$ has a smallest side value of 8, equal to the height of the free space. Thus, the insertion points next to the upper right corner of the rectangle (Coordinate $(9,0)$) marks the least flexible space and is chosen as the best insertion point.

When using the least flexible space to place rectangles in, the number of possible configurations is reduced. The less flexible first algorithm always selects the least flexible space and therefore reduces the possible fields that can be created, and reduces the solution search time.

2.4 Retrieving Data for Analysis

As mentioned, the rectangle packing solver produces solutions that fill the given container. Sets and containers have multiple configurations that show perfect packing. The packing analysis is done by registering if a side of a rectangle touches the edge of the container and which side this is (see: Figure 4). If a side is at the edge of a container the value of the edge is stored as well as the total area of the rectangle. This produces a structured data set ready for processing with the methods described in section 2.5.

2.4.1 Almost Square Rectangle Sets Data Retrieval

In Table 3 an overview is given to show the number of different solution configurations that are produced by the solver. The solution time is given to show the increasing difficulty for

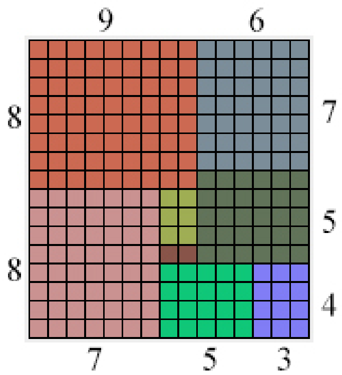


Figure 4: AS-8 solved inside a 15×16 container and the length of the rectangle sides that are at the edge of the container visible.

finding the solutions. The AS-19 and AS-20 sets that are given in Table 3 took over 12 hours to generate all the solutions and therefore were split up into multiple sessions. Because the calculation was split up into multiple sessions it was not possible to give an exact solving time.

2.4.2 Closely Related Rectangle Sets Data Retrieval

Figure 5b shows a configuration that is a solution for a mutated AS-8 rectangle set. The counterpart of the mutated AS-8 rectangle set is given in Figure 5a. In both solutions each rectangle is coloured in the same fashion to show the analogous rectangle in the other solution. The mutated rectangle set solution generates the same data as a non mutated almost square rectangle set but instead of sides in ascending order a larger distribution of sides is present. There also is a possibility of multiple rectangles having the same side size.

The number of closely related rectangle sets that can be created is immense. In section 2.2 the creation of closely related rectangle set is described. Table 4 shows which closely related

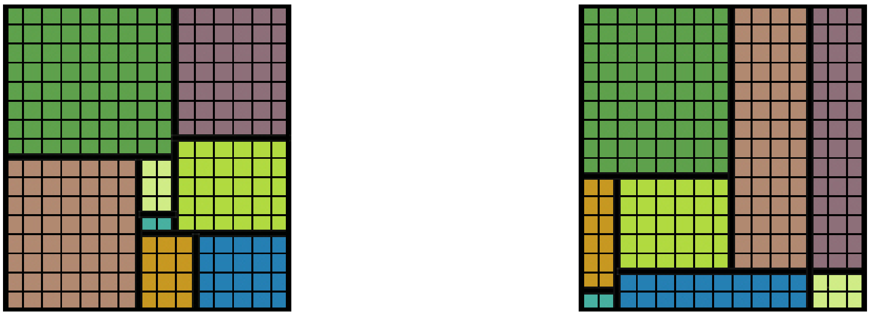
Table 3: Almost Square Rectangle Set and container sizes with the number of solutions in time. A dash in the column means there was no time measurement available because the solving was divided over multiple sessions.

Size N	Container Size	Solutions	Time (s)
11	22×26	16	0,836
13	26×35	168	10,505
14	28×40	12	66,327
	32×35	224	98,858
15	34×40	16	378,751
16	32×51	1492	698,076
17	34×57	64	2990,142
18	30×76	332148	3643,895
19	35×76	1848	-
	38×70	96	-
20	44×70	96	-
	35×88	5496	-

rectangle sets are tested on a given container and how many sets were solvable.

For the AS-11 and AS-13 closely related rectangle sets, all mutations with a factor 2 were analysed. The solving of AS-16 and closely related sets takes a average of 15 minutes. To analyse all the 32768 possible closely related sets of this class would take over a year to calculate on a single processor core. Because of the time restraint, only 255 sets were used in this investigation. As presented in Table 4, the total number of new data points that were created and were analysed with this method was 2124.

The high percentage of number of solutions in the class of closely related sets was a positive surprise. A preliminary survey showed the difficulty of finding solvable sets in perfect rectangle packing problems. The possibility of solving perfect packing for a randomly generated rectangle set with the same area as the container was close to $1 \times 10^{-5}\%$. The high



(a) Solved AS-8 Rectangle Set

(b) Solved Rectangle Set Closely Related to AS-8

Figure 5: Example of Normal AS-8 Rectangle Set in a 15×16 Container and the Closely Related Set Created by using a mutation factor of 2 on the original AS-8 Rectangle Set in the same container.

Table 4: Almost Square Rectangle Set and their closely related rectangle set with container sizes, the total number of sets that was tested, the number of solutions and percentage solvable. Note that the problems of size 16 only a 255 of the total 2^{15} possible sets with a factor of 2 were tested. Sizes 11 and 13 were all tested.

Size N	Container Size	Total Sets	Number Solved	Solve %
11	22×26	1024	309	30.2%
13	26×35	4096	1740	42.5%
16	32×51	255	75	29.4%

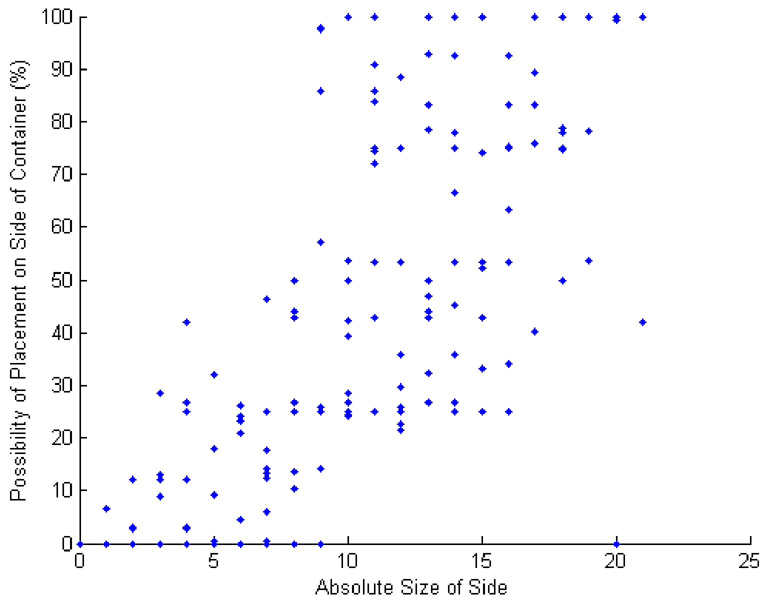
percentage of solvability of the closely related sets ensured a large data set for analysis.

2.5 Analysis of Data

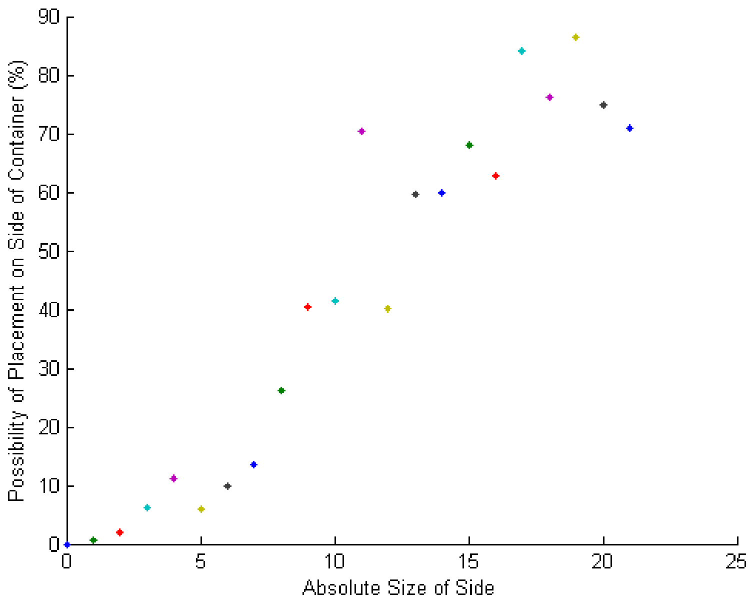
For the analysis of the data a general representation of the data is needed. To compare rectangle sets with different set sizes and rectangle sizes all values were normalized. The normalization is done by taking the value of the largest rectangle side length or area and dividing all other rectangle sides lengths or areas by this value. This generates the relative size percentage of each rectangle in a set and container. The size percentage creates the possibility to compare different sets, containers, and combinations of the former two.

Different sized containers create a different number of solutions for the same rectangle sets, as can be seen in Table 3. For example, the AS-19 packing problem has 96 solutions in a 38×70 container and 1848 solutions for a 35×76 container. To combine the different containers and sets into one diagram a general comparable value is needed. The comparable value for placement of the rectangles on the edge of a container is retrieved with a simple calculation; the total times a rectangle is placed on the edge is divided by the total number of configurations that make up a solution.

When all the processes for generating, normalizing and analysing the data is complete, the raw data is ready to be interpreted to show the results of this thesis.



(a) The possibility of placement on side for a side length.



(b) The average possibility of placement on side by length of side.

Figure 6: AS-14 to AS-20 in their respective containers given in Table 3 plotted to side length. The values are non-normalized. Figure 6a gives all side lengths of all sets in all containers. Figure 6b gives the means of size lengths over all containers.

3 Results

3.1 Almost Square Rectangle Set Analysis

In Figure 6a the placement of rectangle side length is plotted to the probability of placement on the side of the container. There is a correlation visible between all the data points. There are multiple data points per side length, one for each container. By taking the means of each data point with the same side length the data gives a more general depiction for almost square rectangle packing problems. The taking of the means is plotted in Figure 6b.

When normalizing (see: Section 2.5) the results and taking the area of the rectangle into account, it is possible to compare both the area and rectangle side length. This is done in Figure 7. In Figure 7a the correlation between placement and side size is show by the trendline, a linear line. In Figure 7b the fit shows more of a sigmoid function shape with a transition point around 20% in relative area size.

3.2 Closely Related Rectangle Set Analysis

Each result for rectangle sets closely related to the corresponding almost square rectangle set shows the same general trend line as the AS-14 to AS-20 plot (Figure 7). The results are plotted in Figure 8, 9 and, 10. All these plots are normalized and fitted with the same trend line. All trend lines formulas and variables are given in the image description under the image, likewise for the confidence score (R^2).

4 Discussion

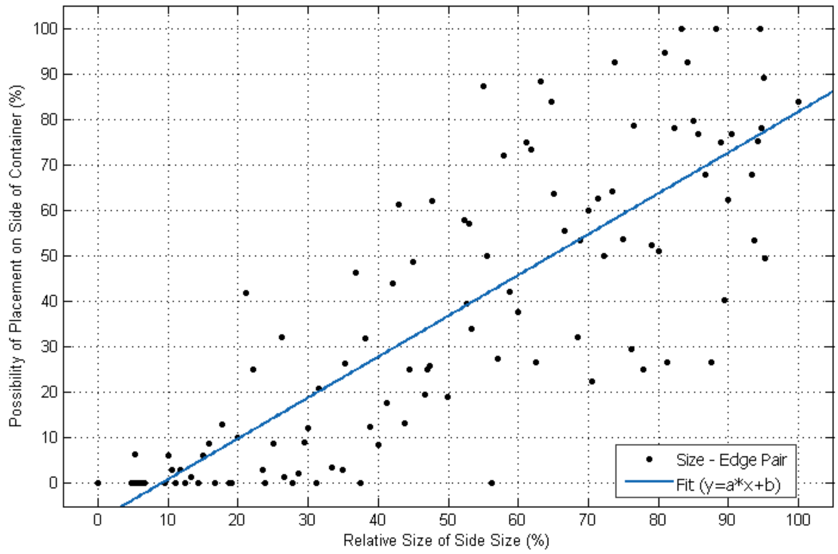
The solving perfect rectangle packing problems proved to be a time consuming task. In previous work, the results of Huang, Korf, Simonis and O‘Sullivan [Huang and Korf, 2012, Simonis and O‘Sullivan, 2011, Korf, 2003] showed that solving these problems could be done in less time than the solver used for this thesis. The results presented in this thesis do give a good indication of all problems and more results would presumably confirm the found outcome.

The finding of the high percentage of solvability of closely related almost square rectangle sets in the containers of their almost square rectangle packing problems was unexpected. The result presented the opportunity to compare different rectangle sets in the same container. The method of generating new rectangle sets gives a new set of problems that can lead to better understanding op packing problems. In section 5.1 the possible usage of this information is discussed.

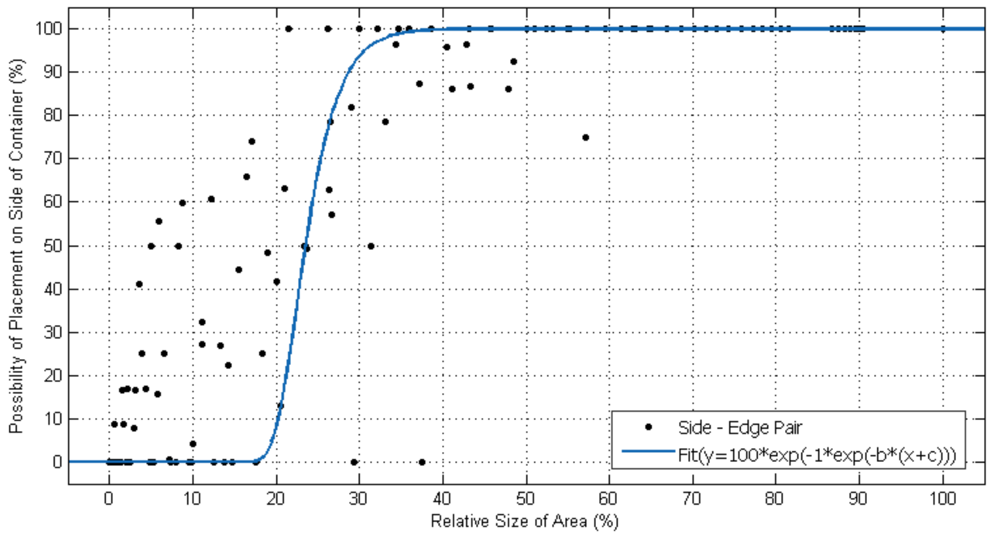
4.1 Possible Explination for Found Results

A possible explanation for the placement of large (in area) rectangles at the edge of the container is that large rectangles *not* in the edge create a space that is not as flexible as the same rectangle placed a the edge. In Figure 11 the space that creates a problem is marked in red. In an almost square rectangle set there are only two rectangles that have a side with size 2, rectangle 1×2 and rectangle 2×3. These two rectangles do not span the total area that is marked in red, thus this configuration can not lead to a solution.

When this rectangle is placed at the edge,

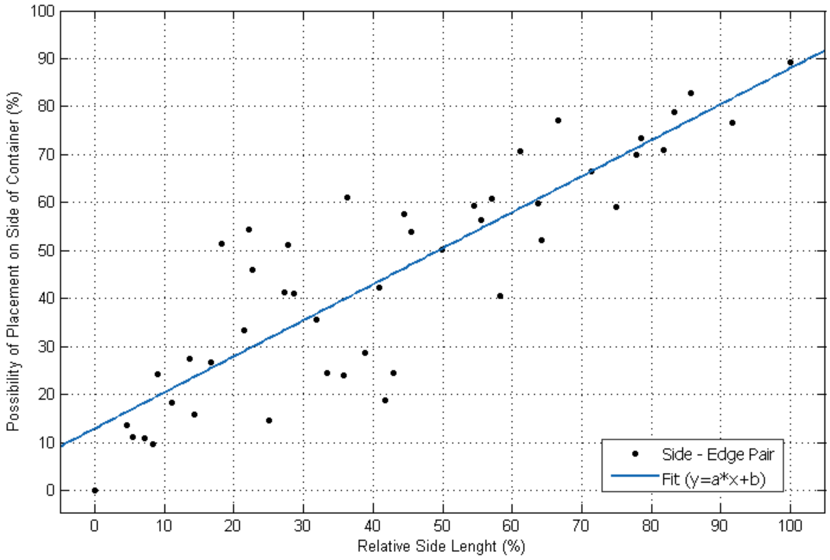


(a) Plot of Placement on edge of container by Relative Lenght of Rectangle Side where $a = 0.8976$, $b = -8.174$ and with R^2 of 0.6611.

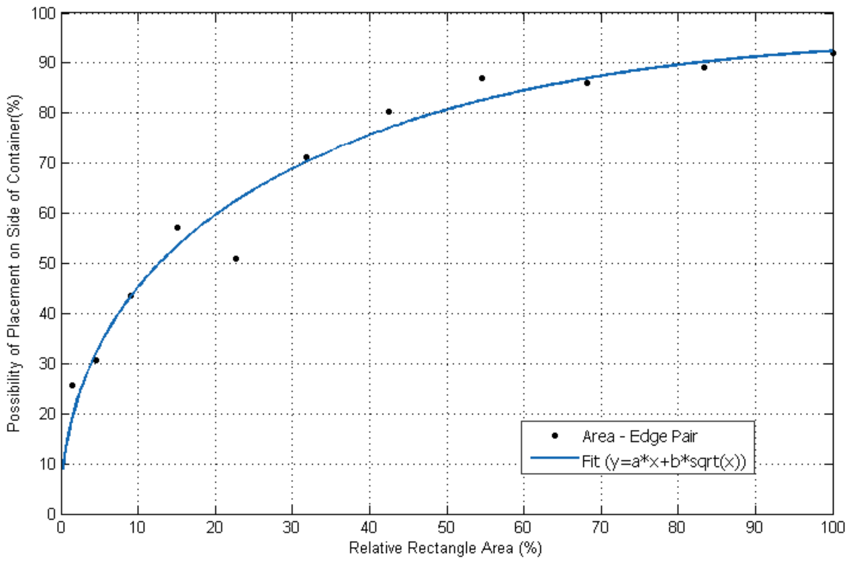


(b) Plot of Placement on edge of container by Relative Area Size where $b = 0.3655$, $c = -22.54$ and with R^2 of 0.9927.

Figure 7: AS-14 to AS-20, non mutated in their respective containers given in Table 3 plotted to side size and area normalized, and means over all configurations of the solutions of the AS-N problem

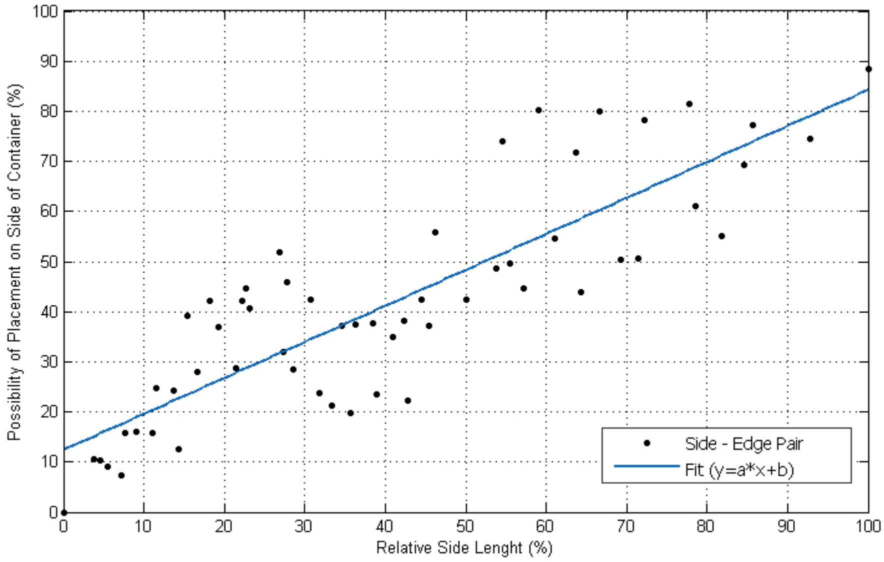


(a) Plot of Placement on edge of container by Relative Lenght of Rectangle Side where $a = 0.7504$, $b = 12.94$ and with R^2 of 0.7513.

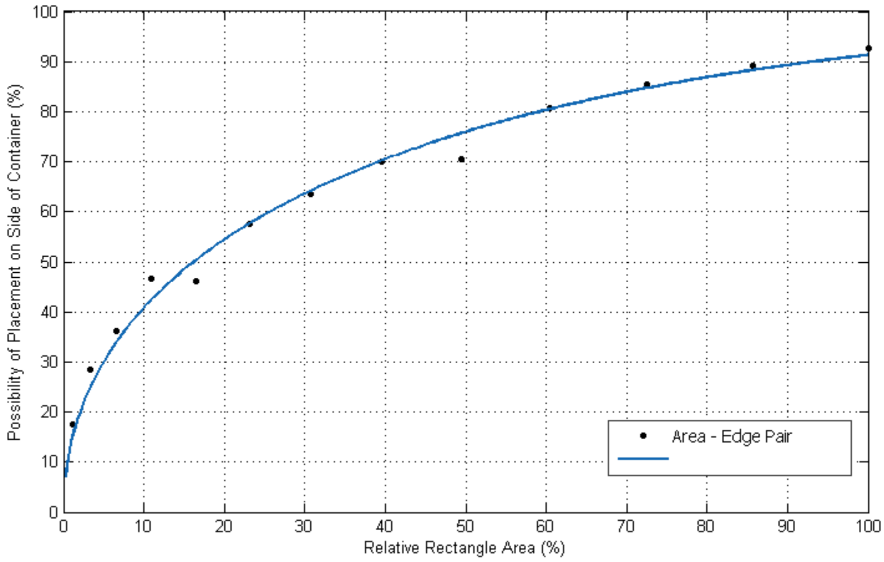


(b) Plot of Placement on edge of container by Relative Rectangle Area Size in Set where $a = -0.7397$, $b = 16.64$ and with R^2 of 0.9584.

Figure 8: Mutated Rectangle Sets Closely Related to AS-11 plotted to side size (Figure 8a) and area (Figure 8b) normalized, and means over all configurations of the solutions in a 22×26 container.

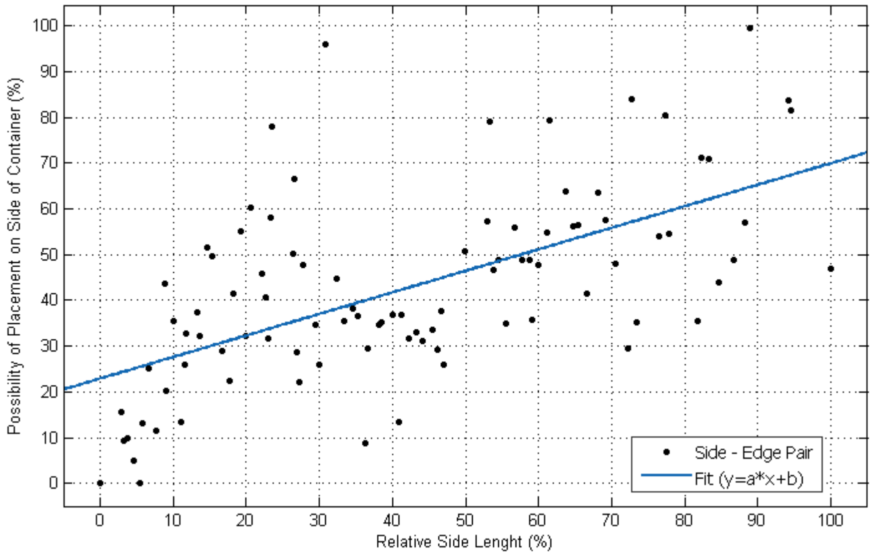


(a) Plot of Placement on edge of container by Relative Lenght of Rectangle Side where $a = 0.7184$, $b = 12.37$ and with R^2 of 0.7231.

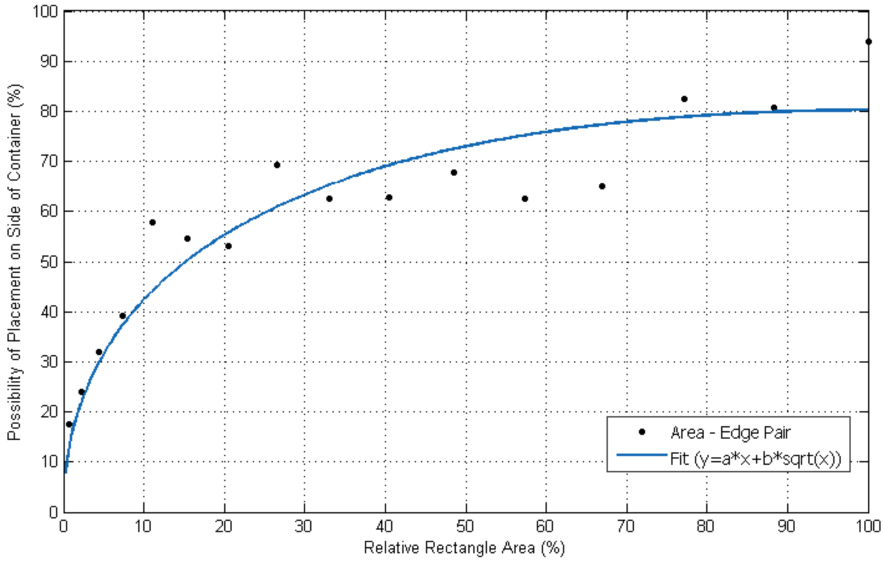


(b) Plot of Placement on edge of container by Relative Rectangle Area Size in Set where $a = -0.5499$, $b = 14.63$ and with R^2 of 0.9857.

Figure 9: Mutated Rectangle Sets Closely Related to AS-13 plotted to side size (Figure 9a) and area (Figure 9b) normalized, and means over all configurations of the solutions in a 26×35 container.



(a) Plot of Placement on edge of container by Relative Lenght of Rectangle Side where $a = 0.4711$, $b = 22.83$ and with R^2 of 0.0.352.



(b) Plot of Placement on edge of container by Relative Rectangle Area Size in Set where $a = -0.7837$, $b = 15.86$ and with R^2 of 0.8636.

Figure 10: Mutated Rectangle Sets Closely Related to AS-16 plotted to side size (Figure 10a) and area (Figure 10b) normalized, and means over all configurations of the solutions in a 32×51 container.

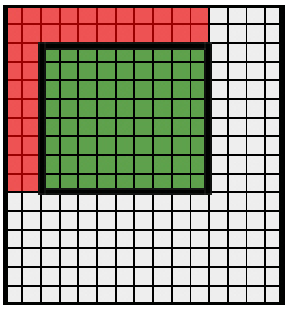


Figure 11: Example of the space that need to be filled (marked in red) when placing a large rectangle not at the edge

the remaining space will not be divided into the problematic space in the previous example. Therefore, it is rational to put large rectangles at the edge.

In the closely related rectangle sets this restraint of small spaces is less of a problem. With a mutation factor of 2, there will be more rectangles that can be packed into the remaining space of Figure 11. A rectangle of 4×5 could be mutated into a rectangle of 2×10 and would fit into the small space. Therefore, the possibility of placement at the edge of a container for large rectangles is not 100 percent but lower. This could explain the difference of placement in almost square rectangle packing and mutated rectangle set packing.

What also could be an explanation for the prevalence of large rectangles at the edge of a container is the flexibility of a smaller rectangle. When rectangles are packed the space that remains will become smaller, and smaller. Rectangles do not fit together perfectly, often small spaces remain that needs to be filled. This space is, more often than not, not at the edge.

4.2 Precision of Rectangle Packer

The precision of the rectangle packer is not confirmed. A number of implementations of packing algorithms showed a number of different solutions. When comparing the results of the rectangle packer of this thesis (Table 3 with the rectangle packer of Simonis et. al. [Simonis and O‘Sullivan, 2011] in Table 5, there are inconsistencies in the number of found solutions. Futhermore, a solver from a third independent party yielded yet another set of results. This seems odd, a seemingly simple rectangle packing problem differs in outcome on three different implementations.

Table 5: Almost Square Rectangle Set and container sizes with the number of solutions as given by Simonis et. al. [Simonis and O‘Sullivan, 2011]

Size N	Container Size	Solutions
11	22×26	4
13	26×35	42
14	28×40	4
15	34×40	4
16	32×51	544
17	34×57	16
18	30×76	110288
19	35×76	526
20	35×88	1988

Simonis et. al. removed all rotations and reflections from their packing solutions, therefore the results from the rectangle packer of this thesis need to be divided by four. The smaller AS-N problems, where N is smaller than 14, the data agrees with Simonis et. al. The inconsistencies with the thesis rectangle packer are for AS-N problems with of size 14, 16, 17, 18, 19 and 20. This is not a problem for this investigation because the solutions that are found with the thesis rectangle packer are correct. It is not needed, but desirable, to find *all* solutions for analysing. It is needed to only include correct solutions and, as many as possible. Because it is not possible to compare all solutions of the current results with the results of Simonis et. al., it is not possible to decide which method is correct. However this could mean that when using this packer for further research, the process of packing needs to be redesigned.

It could be helpful to design a benchmark for perfect packing problems that all rectangle packers have to pass to show correct functioning.

4.3 Reduction of Possible Placements

With the strong indication that large rectangles are frequently placed at the side of a container the number of locations for large rectangles could possibly be reduced.

In Section 2.3 the number of possible locations a rectangle can be placed is given. The number of possible placements of a rectangle in one orientation is as follows:

$$\begin{gathered}(W_{\text{container}} - W_{\text{rectangle}}) \\ \times \\ (H_{\text{container}} - H_{\text{rectangle}})\end{gathered}$$

This equation can now be reduced to

$$\left(\begin{pmatrix}(W_{\text{container}} - W_{\text{rectangle}}) \\ + \\ (H_{\text{container}} - H_{\text{rectangle}})\end{pmatrix} \times 2\right) - 4$$

for each rectangle with a relative large area in one orientation. The number of possible placements in an empty field is thus, drastically reduced. With this given it is possible to test if a set fits into a container. It is not possible to definitely find all solutions for a rectangle set in a container, but the possibility that there is a solution with a large rectangle in the edge when there is a solution at all is very high.

4.4 Low Correlation

In the mutated rectangle sets closely related to AS-16 there is a lower correlation present for both the rectangle side length and the rectangle area. This could be explained by the low number of data points available to create Figure 10. Only the first 255 mutated rectangle sets were used. These rectangle sets do not give a cross-cut of the total possible mutated rectangle sets. When a larger number of the mutated rectangle set would be examined, the results will be less extreme and show a better mean value for all rectangle areas and side lengths.

5 Conclusion

The results all indicate to the same conclusion; rectangles with a large area are predominantly placed on the edge of a container. In the almost square rectangle sets of sizes 14 to 20 (Figure 7b) it is clear that rectangles from the top 50% in area size of a set are almost exclusively placed at the edge. The possibility of placement of these rectangles at the edge are in the range of 99 to 100 percent.

When looking at the mutated sets that are closely related to almost square rectangle sets it is evident that larger rectangles in area are more frequently placed at the edge of a container than smaller rectangles in area size. The largest rectangles in all the evaluated mutated rectangle sets were present at the edge of the container in 90% of the solution configurations.

The possibility of placement at the edge of a container for smaller rectangles drops rapidly for the smallest rectangles.

The side length of a rectangle and placement on the edge of a container shows a weak correlation, but a correlation nonetheless. A larger side has a higher chance of being placed at the edge of a container. But on average the area of a rectangle is a stronger indication than the side length.

5.1 Future Work

With strong evidence for the placement of rectangles with relative large area of a set on the edge, it is possible to create heuristics that exploit this evidence. It is known that rectangle packing problems are time and computation extensive tasks. As mentioned in the introduction it is a NP-complete problem. Instead of examining all possible placements for a large rectangle it is now possible to preferably place the large rectangle at the edge. This could indicate if a rectangle set and a given container have solutions at all. In 90 percent of the time a large rectangle is placed at the side thus intuition tells us that there is only a small chance that *all* solutions for the problem do not have the large rectangle at the side.

In section 2.4.2 the number of rectangle sets that are solvable is given for AS-11, AS-13 and a selection of AS-16. This process was time consuming and with the new created heuristic could reduce this time for investigating if the created rectangle set has a solution in the container. If there is a solution found with the new heuristic it is justified to check all possible configurations for a solution.

References

[Dowsland, 1987] Dowsland, K. A. (1987). An exact algorithm for the pallet loading problem. *European Journal of Operational Research*, 31(1):78 – 84.

[Huang and Korf, 2012] Huang, E. and Korf, R. E. (2012). Optimal rectangle packing: an absolute placement approach. *Journal of Artificial Intelligence Research*, 46:47–87.

[Korf, 2003] Korf, R. E. (2003). Optimal rectangle packing: Initial results. In *ICAPS*, pages 287–295.

[Onodera et al., 1991] Onodera, H., Taniguchi, Y., and Tamaru, K. (1991). Branch-and-bound placement for building block layout. In *Proceedings of the 28th ACM/IEEE Design Automation Conference*, pages 433–439. ACM.

[Perdeck, 2011] Perdeck, M. (2011). Fast optimizing rectangle packing algorithm for building css sprites.

[Simonis and O'Sullivan, 2011] Simonis, H. and O'Sullivan, B. (2011). Almost square packing. In *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, pages 196–209. Springer.

[Simonis and OSullivan, 2008] Simonis, H. and OSullivan, B. (2008). Search strategies for rectangle packing. In *Principles and Practice of Constraint Programming*, pages 52–66. Springer.

[Wu et al., 2002] Wu, Y.-L., Huang, W., Lau, S.-c., Wong, C., and Young, G. H. (2002). An effective quasi-human based heuristic for solving the rectangle packing problem. *European Journal of Operational Research*, 141(2):341–358.