Ani Megerdichian

# Bin Packing

## Introduction

In this report, we will test the performance of five algorithms that aim to solve the bin packing problem. For these experiments, the bin packing problem is one where there are N items with weights or sizes that range between 0 and 0.6. The objective is to fit these items into the smallest number of bins, where the sum of the weights/sizes of the items in each bin cannot exceed 1. We will be calculating the waste each algorithm produces for increasing input sizes. Per the assignment specifications, the waste, W(A), of a bin packing algorithm, A, for any given list of items, is the number of bins used by the algorithm A minus the total size of all the items in the list. The respective bin packing algorithms are Next Fit, First Fit, First Fit Decreasing, Best Fit, and Best Fit Decreasing.

## Zip Tree

All but the Next Fit algorithm use a Zip tree for their implementation. A Zip tree is a binary search tree where each node has a rank that is retrieved from a geometric distribution. The tree is in max heap order with respect to these ranks. The implementation of the Zip tree used for this project was guided by the paper *Zip Trees* by Tarjan et al. (https://www.ics.uci.edu/~goodrich/teach/cs165/notes/ZipTrees.pdf).

The reason for using a Zip tree to implement the First Fit, First Fit Decreasing, Best Fit, and Best Fit Decreasing algorithms is that these implementations run in O(nlogn) time instead of simpler implementations that run in O($n^2$) time.

## Data Collection

Each algorithm was tested using inputs of increasing sizes with items of weights between 0.0 and 0.6 that are generated uniformly at random. For each input size, a new vector of items is generated and tested with each algorithm for 15 repetitions. The average waste value for these 15 repetitions is taken as a data point for the given algorithm. Input sizes start at $2^1$ and increase by a multiple of 2 until a trend in the waste values is observed and sufficient data is collected.

# Bin Packing Algorithms and Determining Waste Experimentally

## 1.Next Fit

In the Next Fit algorithm, if the given item fits into the current bin, the item is placed in that bin. Otherwise, the item goes into a new bin. This is implemented using a vector where the algorithm keeps track of the current bin or index in the vector. This algorithm runs in O(n) time.

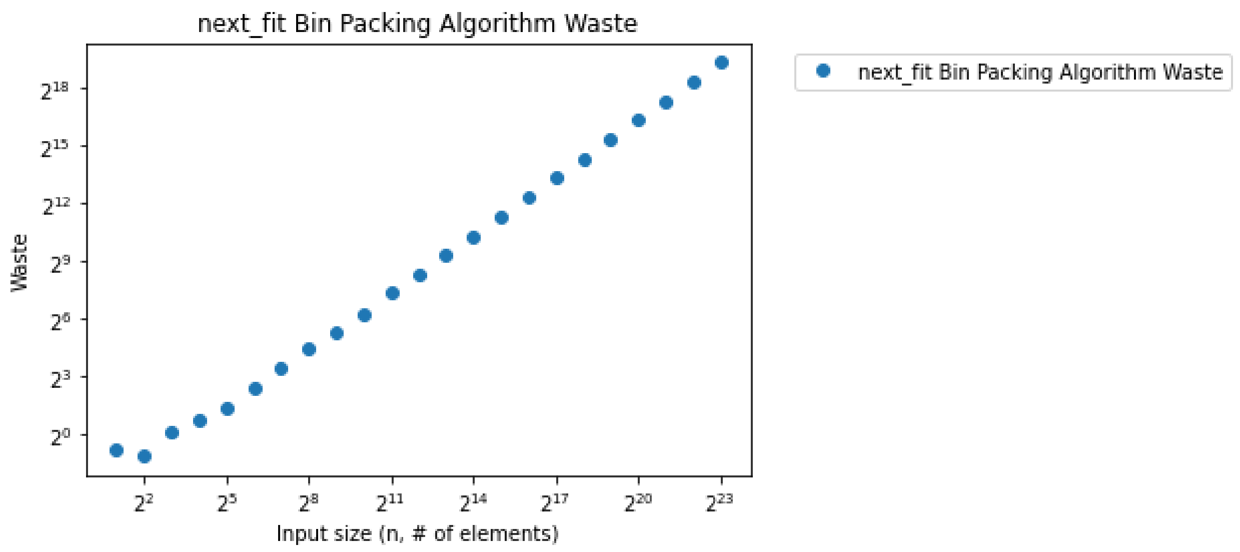


*Figure 1.1 shows the log-log graph of the data collected for the next_fit bin packing algorithm with the input size as the x-axis and the calculated waste as the y-axis when given uniformly at random item weights between 0.0 and 0.6.*

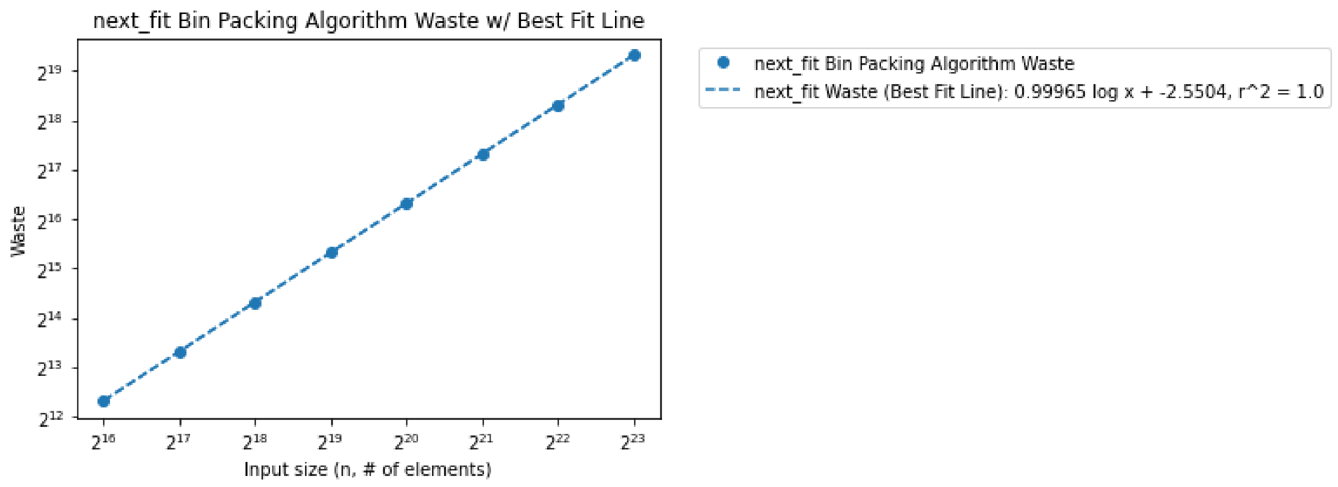


*Figure 1.2 shows the log-log graph of the data collected for the next_fit bin packing algorithm with the input size as the x-axis and the calculated waste as the y-axis when given uniformly at random item weights between 0.0 and 0.6 along with its accompanying best fit line. Note that data points with an input size less than $2^{16}$ were removed in determining the best fit line for the graph in order to reduce the noise in the best fit line determination.*

As seen in Figure 1.2, the best fit line for the waste of the next_fit bin packing algorithm is 0.99965log(x) + -2.5504 where the slope is 0.99965, the y-intercept is -2.5504 and the coefficient of determination is $r^2 = 1.0$. Therefore, the waste produced using the next fit algorithm is $O(n^{1.0001})$.

## 2. First Fit

The First Fit algorithm works by inserting each given item into the first bin that has enough space to hold the item. If a bin to hold the item is not found, a new bin is created. This implementation uses a Zip tree where each node represents a bin and has a rank assigned randomly from a geometric distribution, a key which is its bin number, and a pair of values that are the node's remaining capacity, and the best remaining capacity of its subtrees. For each item, an inorder traversal is done until the node that has a remaining capacity greater than or equal to the item is found. If such node is not found, a new node is inserted into the tree, representing the addition of a new bin. The run time of this algorithm is O(nlogn).

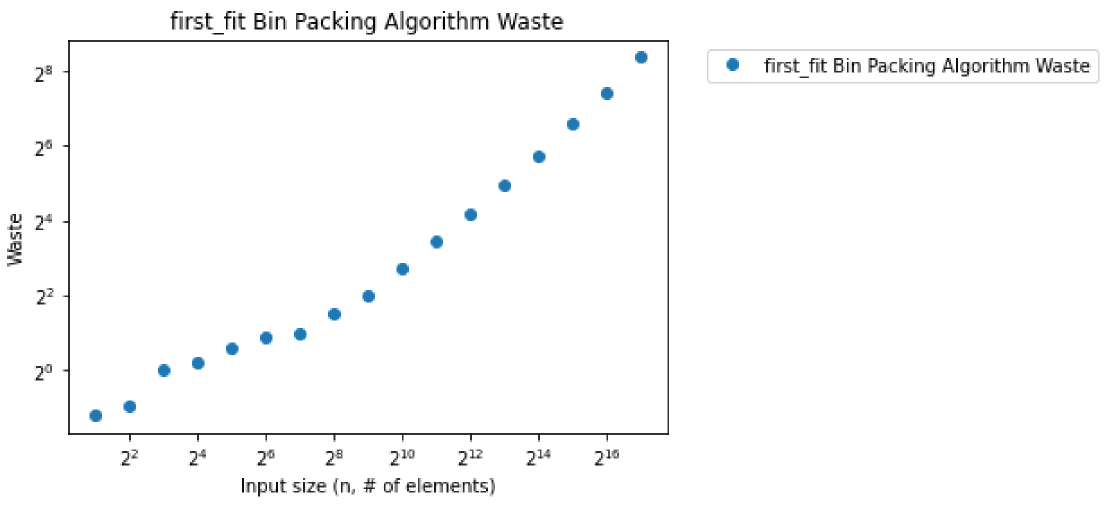


*Figure 2.1 shows the log-log graph of the data collected for the first_fit bin packing algorithm with the input size as the x-axis and the calculated waste as the y-axis when given uniformly at random item weights between 0.0 and 0.6.*

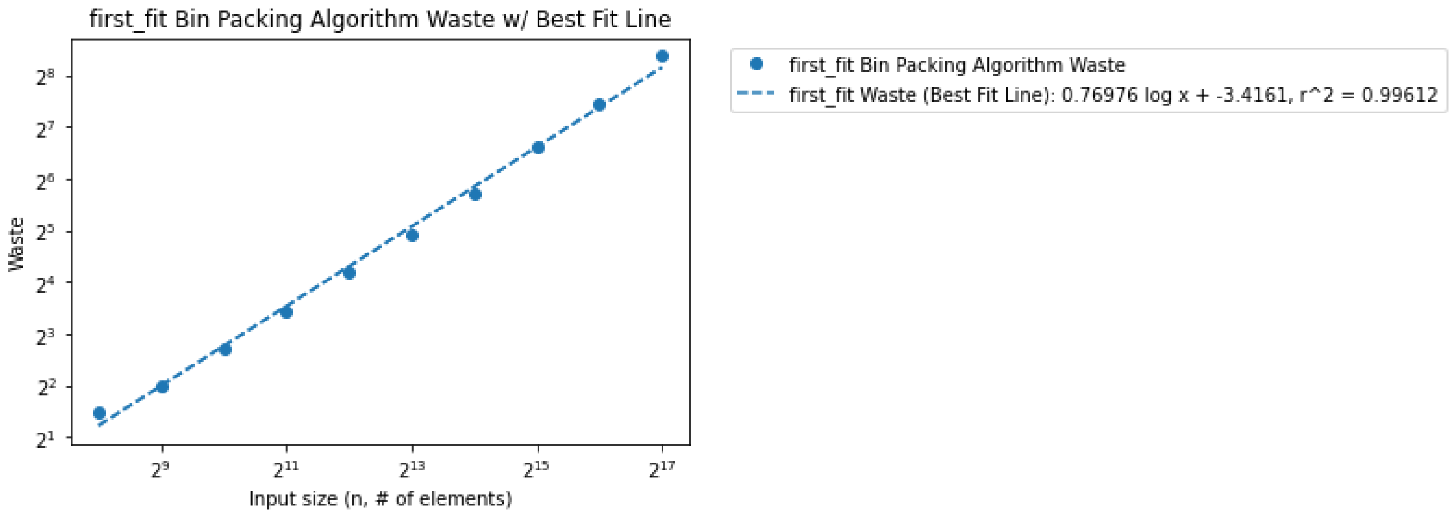


*Figure 2.2 shows the log-log graph of the data collected for the first_fit bin packing algorithm with the input size as the x-axis and the calculated waste as the y-axis when given uniformly at random item weights between 0.0 and 0.6 along with its accompanying best fit line. Note that data points with an input size less than $2^8$ were removed in determining the best fit line for the graph in order to reduce the noise in the best fit line determination.*

As seen in Figure 2.2, the best fit line for the waste of the first_fit bin packing algorithm is 0.76976log(x) + -3.4161 where the slope is 0.76976, the y-intercept is -3.4161 and the coefficient of determination is $r^2 = 0.99621$. Therefore, the waste produced using the next fit algorithm is $O(n^{0.76976})$.

## 3. First Fit Decreasing

The First Fit Decreasing algorithm is identical to the First Fit algorithm, except the vector of items is first sorted in descending order of size before the items are placed into bins.

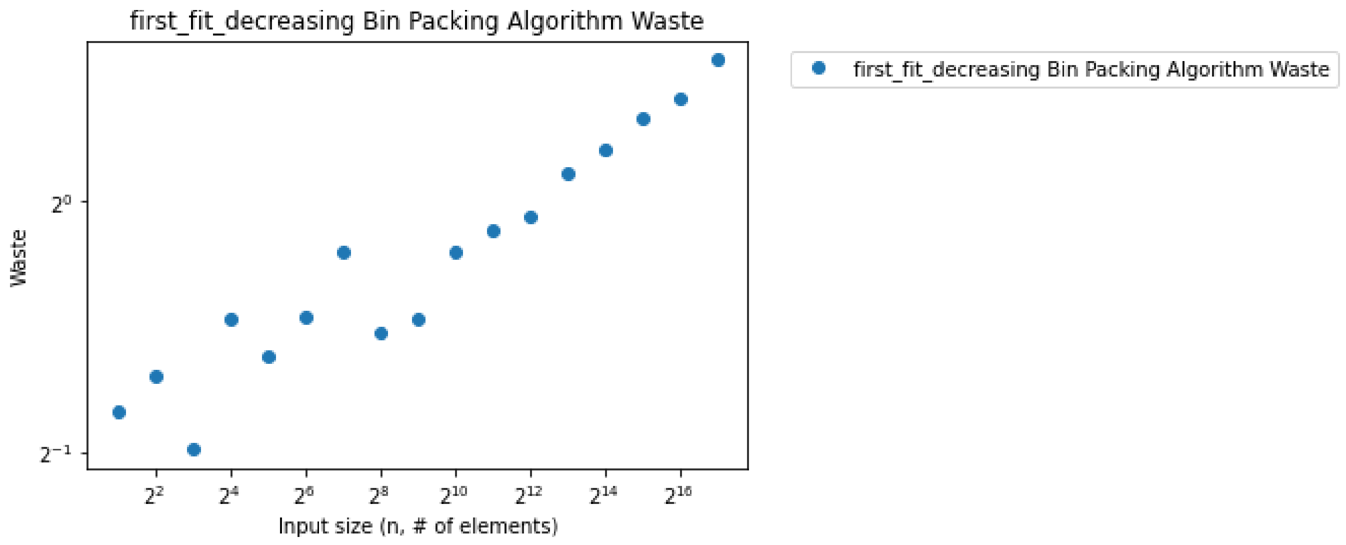


*Figure 3.1 shows the log-log graph of the data collected for the first_fit_decreasing bin packing algorithm with the input size as the x-axis and the calculated waste as the y-axis when given uniformly at random item weights between 0.0 and 0.6.*

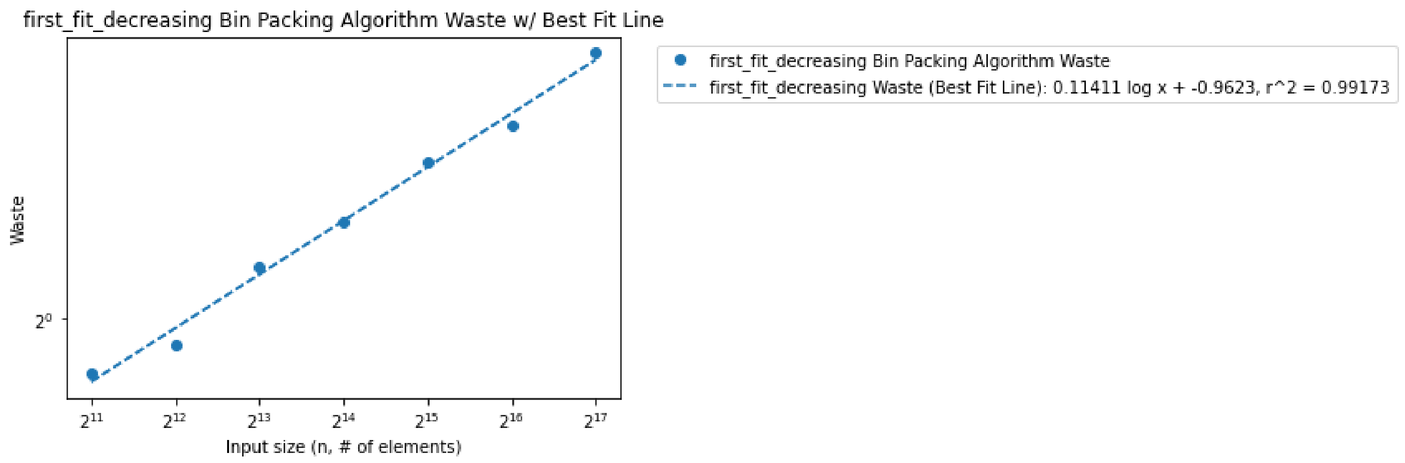


*Figure 3.2 shows the log-log graph of the data collected for the first_fit_decreasing bin packing algorithm with the input size as the x-axis and the calculated waste as the y-axis when given uniformly at random item weights between 0.0 and 0.6 along with its accompanying best fit line. Note that data points with an input size less than $2^{11}$ were removed in determining the best fit line for the graph in order to reduce the noise in the best fit line determination.*

As seen in Figure 3.2, the best fit line for the waste of the first_fit_decreasing bin packing algorithm is 0.11411log(x) + -0.9623 where the slope is 0.11411, the y-intercept is -0.9623 and the coefficient of determination is $r^2 = 0.99173$. Therefore, the waste produced using the next fit algorithm is $O(n^{0.11411})$.

## 4. Best Fit

The Best Fit algorithm places each item into the bin that has the smallest remaining capacity greater than or equal to the item size. In other words, it places the item into the bin where it fits the tightest. If no such bin exists, a new bin is created, and the item is put into that bin. This algorithm is implemented using a Zip tree where each node has a rank, a key that denotes a remaining bin capacity, and a vector value that holds all of the bin numbers that have the node's respective remaining capacity. An inorder traversal is done for each item to find the node with the smallest remaining capacity that is greater than or equal to the item's size. If such node is found, one of the bins associated with that node is assigned to that item. Otherwise, a new bin is created. This algorithm runs in O(nlogn) time.

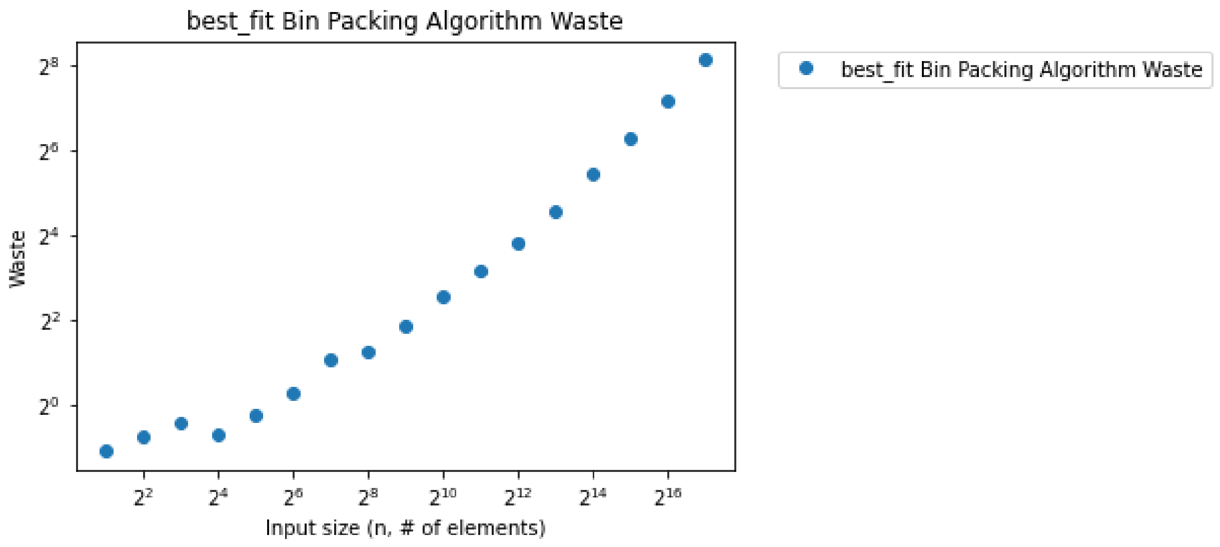


*Figure 4.1 shows the log-log graph of the data collected for the best_fit bin packing algorithm with the input size as the x-axis and the calculated waste as the y-axis when given uniformly at random item weights between 0.0 and 0.6.*

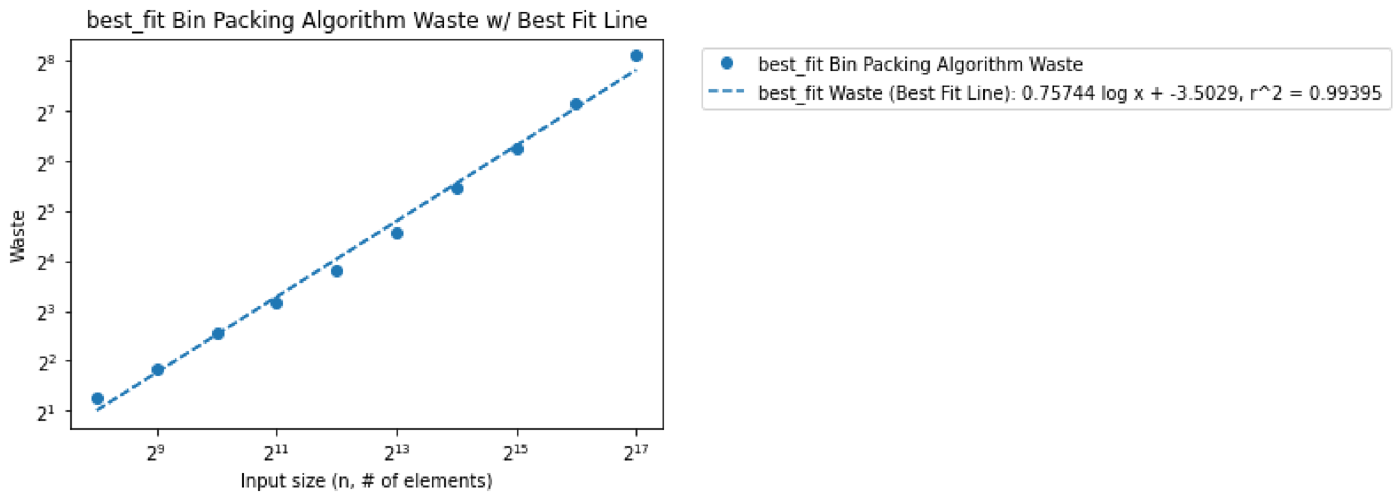


*Figure 4.2 shows the log-log graph of the data collected for the best_fit bin packing algorithm with the input size as the x-axis and the calculated waste as the y-axis when given uniformly at random item weights between 0.0 and 0.6 along with its accompanying best fit line. Note that data points with an input size less than $2^8$ were removed in determining the best fit line for the graph in order to reduce the noise in the best fit line determination.*

As seen in Figure 4.2, the best fit line for the waste of the best_fit bin packing algorithm is 0.75744log(x) + -3.5029 where the slope is 0.75744, the y-intercept is -3.5029 and the coefficient of determination is $r^2 = 0.99395$. Therefore, the waste produced using the next fit algorithm is $O(n^{0.75744})$.

## 5. Best Fit Decreasing

The Best Fit Decreasing algorithm is identical to the Best Fit algorithm, except the vector of items is first sorted in descending order of size before the items are placed into bins.

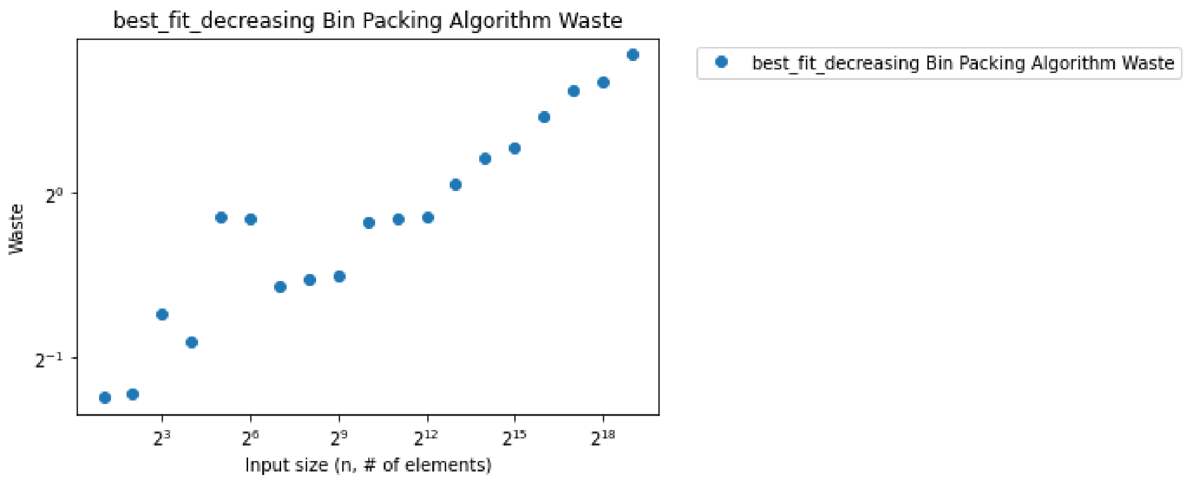


*Figure 5.1 shows the log-log graph of the data collected for the best_fit_decreasing bin packing algorithm with the input size as the x-axis and the calculated waste as the y-axis when given uniformly at random item weights between 0.0 and 0.6.*

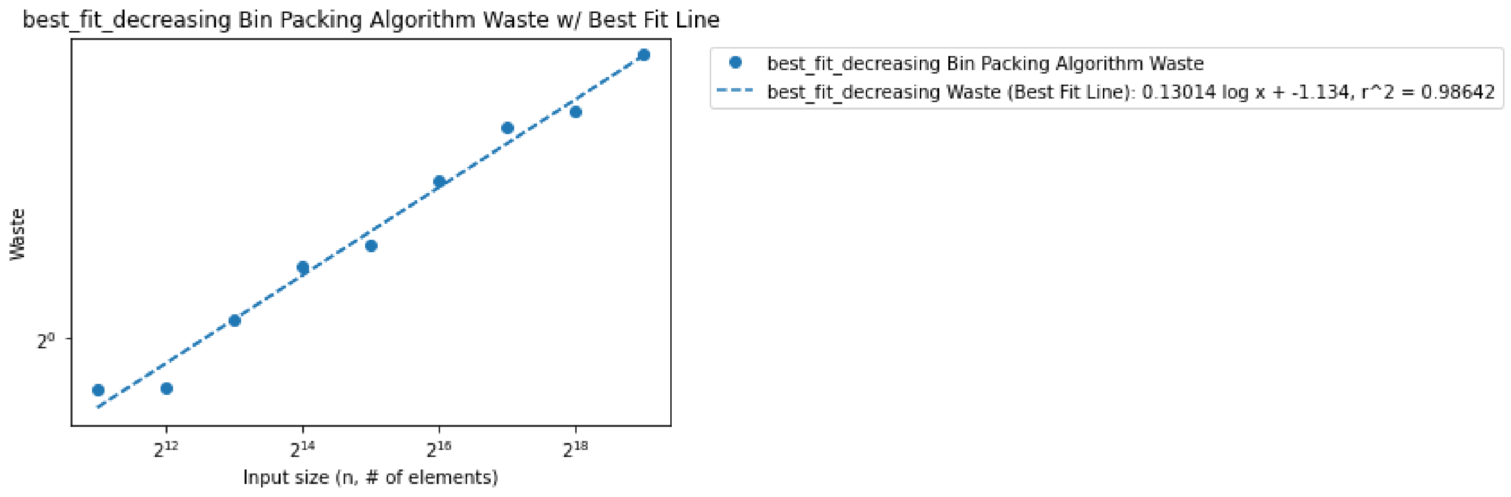
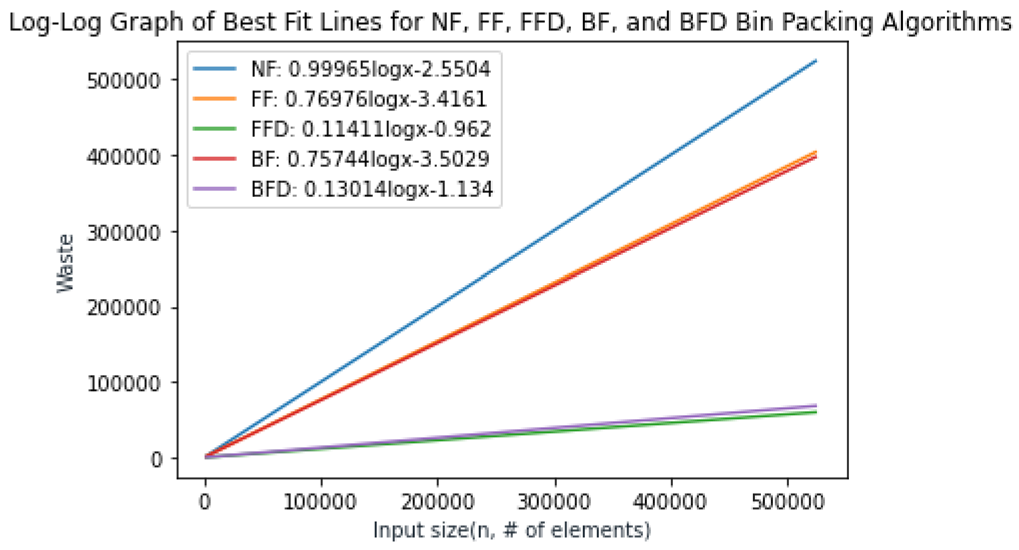


*Figure 5.2 shows the log-log graph of the data collected for the best_fit_decreasing bin packing algorithm with the input size as the x-axis and the calculated waste as the y-axis when given uniformly at random item weights between 0.0 and 0.6 along with its accompanying best fit line. Note that data points with an input size less than $2^{11}$ were removed in determining the best fit line for the graph in order to reduce the noise in the best fit line determination*

As seen in Figure 5.2, the best fit line for the waste of the best_fit_decreasing bin packing algorithm is 0.13014log(x) + -1.134 where the slope is 0.13014, the y-intercept is -1.134 and the coefficient of determination is $r^2 = 0.98642$. Therefore, the waste produced using the next fit algorithm is $O(n^{0.13014})$.

## Conclusion: Best Bin Packing Algorithm



The figure above shows the log-log graph of the best fit lines determined for each of the five bin packing algorithms as described in the previous sections. The slope of the best fit line for the first_fit_decreasing algorithm is the smallest. This means that as the input size, or number of items increases, the waste produced by the first_fit_decreasing algorithm increases at a slower rate compared to the other four algorithms. ***For this, I deem the first_fit_decreasing algorithm as the best of the bin packing algorithms discussed in this report.***

One issue in bin packing is finding bins for items of large sizes or weights, especially when they appear later on in the list of items. By first sorting the items from largest to smallest, as the first_fit_decreasing algorithm does, we can remedy this issue. By sorting the items in this manner, you are able to get a 'tighter' fit in the bins. Every time you look to fit an item into a bin, you know that there are no other items larger than that item. In addition, sorting allows you to allocate larger items to bins, and then fill out those bins with smaller sized items which decreases the total waste.

For example, take items [0.6, 0.3, 0.7, 0.4] to illustrate this point.

1. *With the first_fit algorithm, or best_fit algorithm, the bin allocations would be:*
   *Bin 1: 0.6, 0.3*
   *Bin 2: 0.7*
   *Bin 3: 0.4*
   *Waste = 1*
2. *With the first_fit_decreasing algorithm, the bin allocations would be:*
   *Bin 1: 0.6, 0.4*
   *Bin 2: 0.7, 0.3*
   *Waste = 0*

In the second case, we are able to take advantage of the fact that the pair of items (0.6, 0.4) and (0.7, 03) can be put into the same bins to take up all the space in the bins by first sorting the items in decreasing order. The first case does not do this which results in a larger waste value. The best_fit_decreasing algorithm works in a similar manner, and as the figure above shows, its waste is $O(n^{0.13014})$, which is similar to that of fit_first_decreasing.