Hoan Tran

20 May 2019

CS165 Project 2 Report: The Bin Packing Problem

Overview:

The bin packing problem asks how to optimally pack several equal sized bins with various items of different weights or volume. The optimal solution to this problem would be to minimize the number of bins needed to full pack every item. Unfortunately, there has not been any algorithms to find the optimal solution so far since there will exist some set of parameters or inputs that would give an nonoptimal solution for each algorithm. Although the optimal solution can not be guaranteed, we can still try to find the best nonoptimal solution. This report will analyze each algorithm to find the best overall algorithm even if it is not optimal. The algorithms that will be tested are the Next Fit, First Fit, Best Fit, First Fit Decreasing, and Best Fit Decreasing algorithms.

Testing:

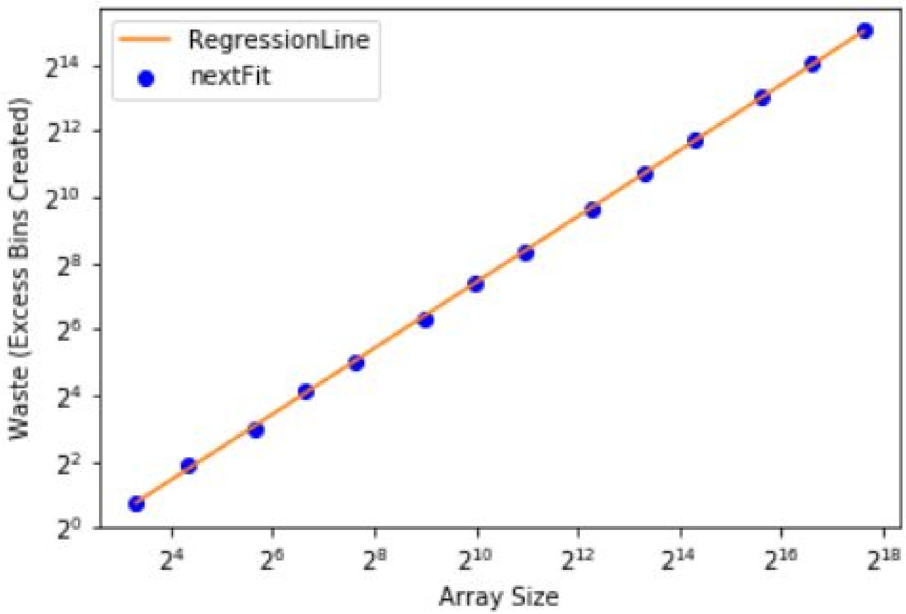
The tests will check for Waste(N) which will be the number of excess bins as a function of N, the number of items that are being packed. Each algorithm will be run multiple times and averaged for each input size. The averages for each input size will then be plotted on a loglog scale (base 2) so as to find a linear regression that represents how quickly each algorithms' Waste scales with input size.

Algorithms:

1. Next Fit Algorithm
    a. Description: The Next Fit Algorithm will only keep track of the last bin that was created and try to fit items into the current bin. This reduces any overhead from trying to remember every bin created thus far, but is obviously not optimal since there may exist an earlier bin to fit later items that the algorithm does not check for.
    b. Pseudocode:

```
for each item:
    if the item can fit inside the current bin:
        put it inside the current bin
    else:
        make a new bin and place the item inside
        current bin = the new bin
```

c. Results:




```
Regression Line Function: 0.9967522070692278x + -2.548293824671309
```

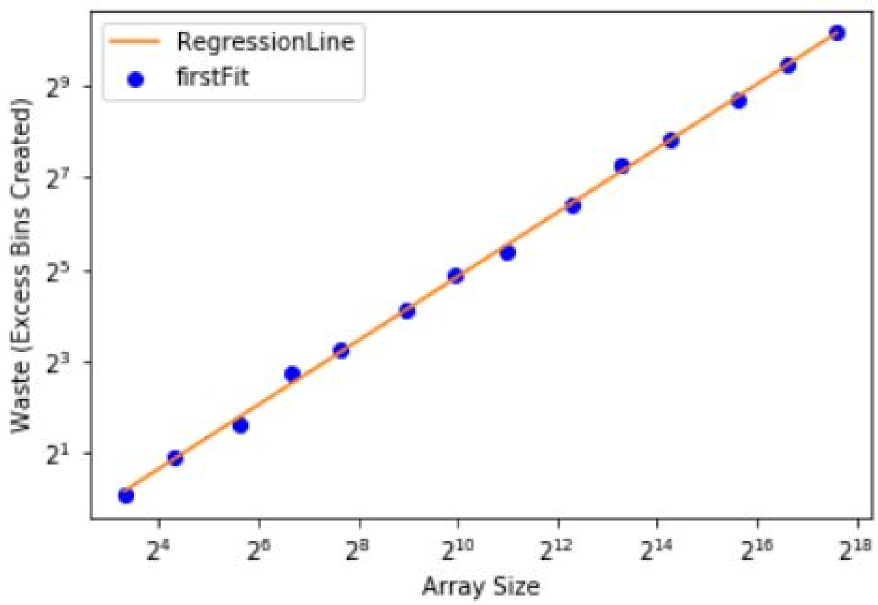
2. First Fit Algorithm

   a. Description: The First Fit algorithm keeps track of every bin that has been created and checks for the first bin that can fit the item. This algorithm is not optimal because an item might be placed into a bin that appears before the optimal bin it should go into. However, it should still be better with waste the Next Fit because on average it should be creating less bins than Next Fit for each item. There is an implementation of the algorithm that reduces the overhead of checking each bin that can fit an item that uses a tree-like structure, but I did not implement it for this report.

b. Pseudocode:

```
for each item:
    for each bin:
        if the item fits in the bin:
            Place the item inside the bin
            Move onto the next item
    if the item can not fit into any bins:
        make a new bin and place the item inside
```

c. Results:



Regression Line Function: 0.6967529434886589x + -2.1239458353855967

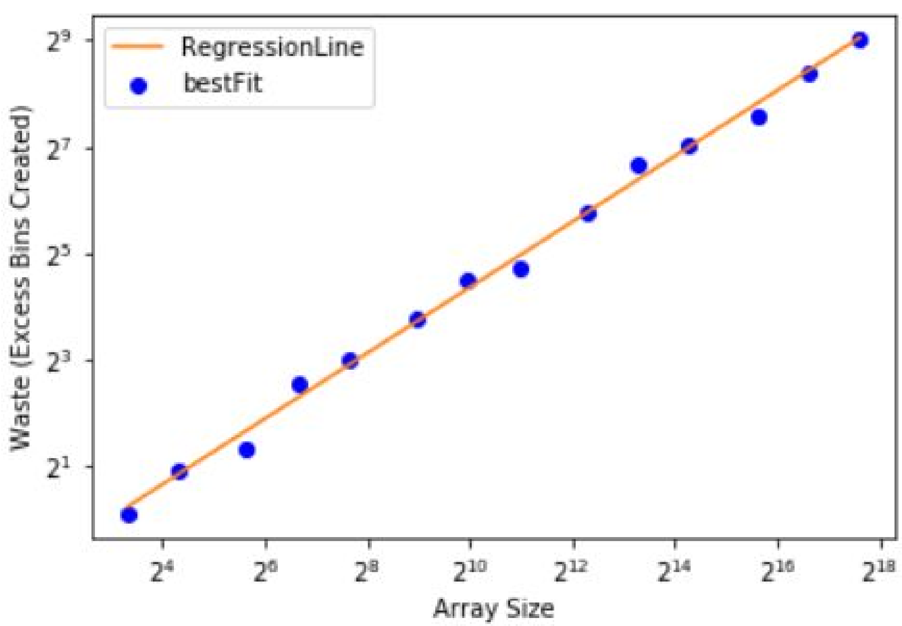
3. Best Fit Algorithm

   a. Description: The Best Fit algorithm also keeps track of every allocated bin. However, it now checks where the item "best fits" into a bin, meaning that the empty space inside a bin is the closest to the item's weight or volume. This algorithm tries to find the best spot for each item. Unfortunately, the items are checked independently so there may be an optimal combination of items that should be packed together, but are split into different bins because of the order in which the items are added. Similar to First Fit, there is an implementation that uses a tree-like structure to reduce the time to check for the best bin which I also did not implement here.

b. Pseudocode:

```
for each item:
      for each bin:
            find the bin that can fit the item and has
            the smallest free space available
      if the smallest bin is found:
            Place item into that smallest bin
      if the item can not fit into any bins:
            make a new bin and place the item inside
```

c. Results:



Regression Line Function: 0.615684129415483x + -1.800988657007006

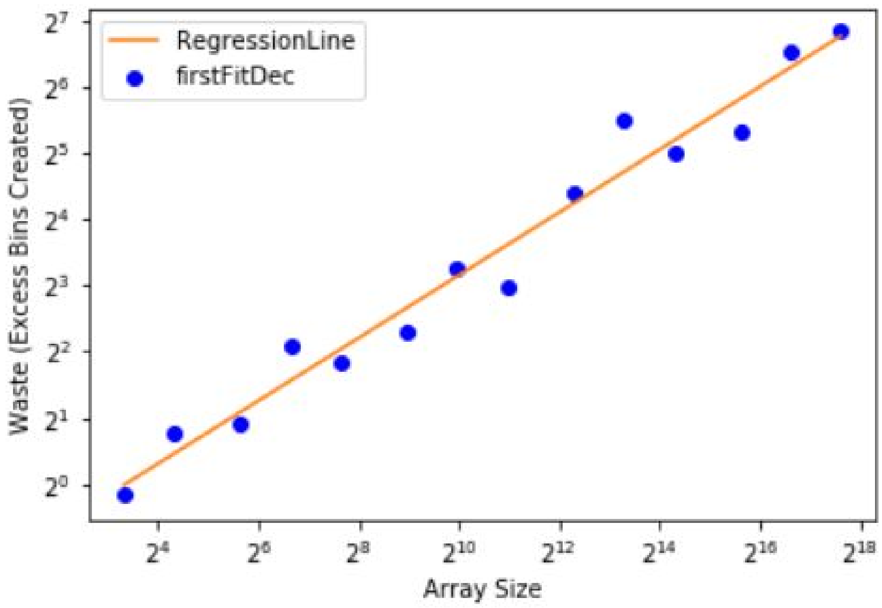
4. First Fit Decreasing Algorithm

    a. Description: The First Fit Decreasing algorithm is exactly the same as the regular First Fit algorithm except that the input array of items is now sorted in decreasing order. The reasoning behind that is to place the larger items separately from each other so that more of the smaller items can be placed in the bins with the larger items. This algorithm can also use the search tree implementation that the regular First Fit algorithm can use, but I did not implement it for these tests.

    b. Pseudocode:

```
sort the list of items in decreasing order
for each item:
    for each bin:
        if the item fits in the bin:
            Place the item inside the bin
            Move onto the next item
    if the item can not fit into any bins:
        make a new bin and place the item inside
```

c. Results:




```
Regression Line Function: 0.4747221948076451x + -1.5922048059743468
```

5. Best Fit Decreasing Algorithm

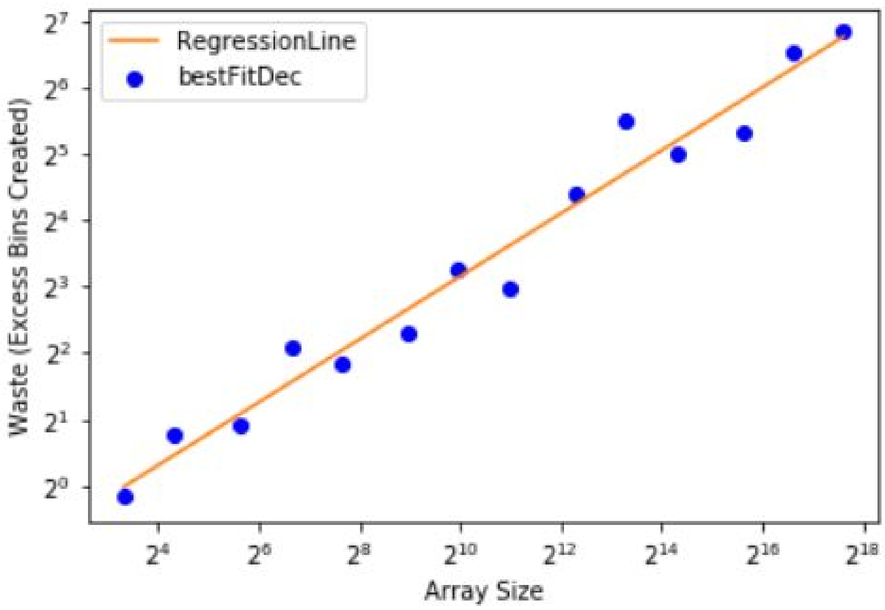
   a. Description: The Best Fit Decreasing algorithm is exactly the same as the regular Best Fit algorithm except that the input array of items is sorted by decreasing order. The reasoning is the same as why the First Fit Decreasing algorithm also sorts its input. This algorithm can also use the search tree implementation that the regular Best Fit algorithm can use, but I did not implement it for these tests.

   b. Pseudocode:

```
sort the list of items in decreasing order
for each item:
      for each bin:
            find the bin that can fit the item and has
            the smallest free space available
      if the smallest bin is found:
```

```
            Place item into that smallest bin
        if the item can not fit into any bins:
            make a new bin and place the item inside
```

c. Results:




```
Regression Line Function: 0.4746012938794427x + -1.5915707629095435
```

Conclusion:

Based on the regression lines for each algorithm, the Best Fit Decreasing algorithm seems to be the best algorithm because its slope coefficient is the least which means the amount of Waste scales the best with larger input sizes out of all of the algorithms. However, it is only marginally better the the First Fit Decreasing algorithm meaning that the First Fit Decreasing algorithm might also be a valid option to use. The worst algorithm by far is the Next Fit algorithm which was to be expected since it is a kind of "dumb" bin packing algorithm that doesn't remember any of its previous bins.