Chenlin Li

Bin Packing Report

Introduction

Bin packing problem is the problem about the different size of objects be packed into finite number of bins. “It is a NP-hard problem, and it is NP-complete about deciding if all objects can fit into specific bins.” (“Bin packing problem”) There are many application using the Bin packing problem, such as “ filling up containers, loading trucks with weight capacity constraints, creating file backups in media and technology mapping in field-programmable gate array semiconductor chip design.” (“Bin packing problem”)

In this research, I implemented five bin packing algorithms to test the quality of the solutions they produced. The five bin- packing algorithms are Next Fit, First Fit, Best Fit, First Fit Decreasing, and Best Fit Decreasing. I created a lot of random double numbers (from 0 – 1) to test the quality of each algorithm. Besides, I used the collected data and the graph which draw based on the data to show the result of each bin-packing algorithm. I compared those algorithms to find which one had better quality.

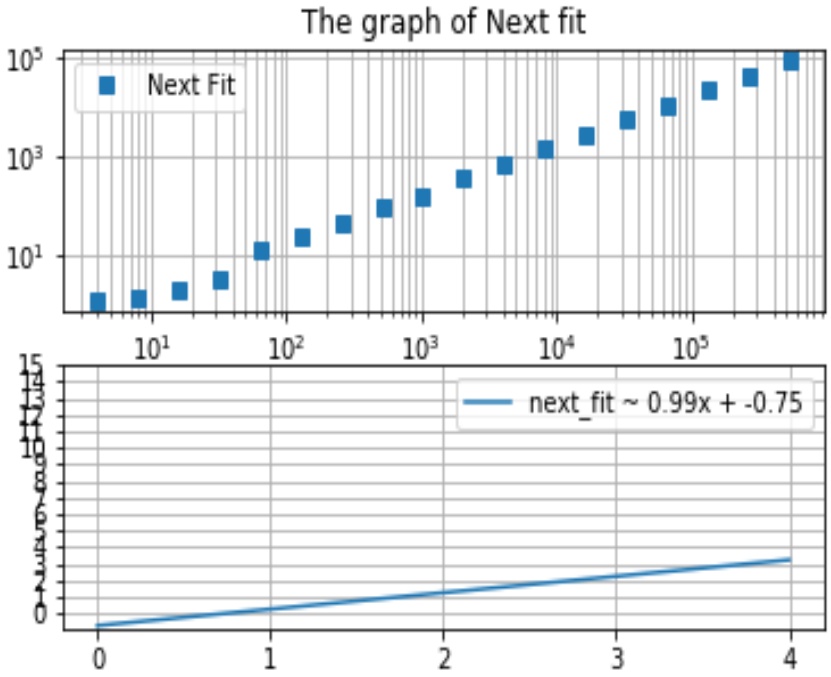
Next Fit

Pseudo Code

```
Random items vector, V = (V1, V2, …, Vn)
Assignment vector, A = (V.size(),0)
Free space vector, F is an empty vector
Function next_fit (Vector V,  Vector A, Vector F){
        Let B initialize to 0 which is the number of the current bin's loading
        Let N initialize to 0 which is the bin number
        For i = 1 to n do:
                If  B + V[i] <= 1 then
                        Add the V[i] to the B
                        Assign the A[i] is N
                Else then
                        Push back the remain of the bin(1 – B) into the F
                        Set B is V[i]
                        Update N will adding one to N
                        Assign the A[i] is N
        If the B + V[n] <=1 then
                Push back the remain of the bin (1-B) into the F
}
```

Result of the Next fit

test_size	waste
4	1.21683
8	1.44903
16	1.86393
32	3.45528
64	12.8203
128	23.4405
256	45.3784
512	91.0153
1024	159.488
2048	354.76
4096	686.071
8192	1373.31
16384	2719.9
32768	5437.06
65536	10951.8
131072	21814.1
262144	43509.6
524288	87415.8



Analysis of the Result

From the above data and graph, I could get it was a linear function, and its function was y = 0.99x + -0.75. The Next Fit run very fast, but it could cause many wastes. In a very large storage, the Next Fit could be a good way to be used. However, if we wanted to save storage, we should use other ways to pack bins.

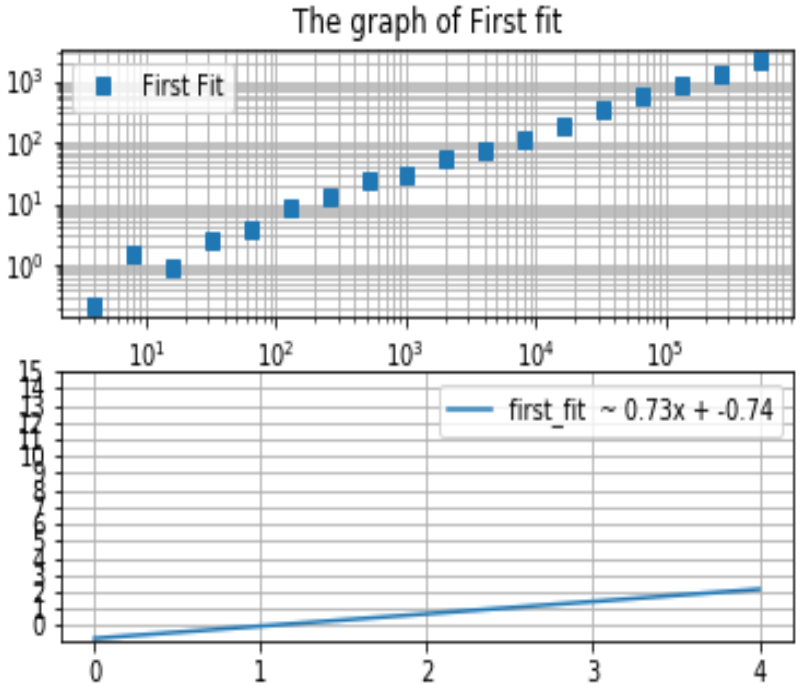
First Fit

Pseudo Code

```
Random items vector, V = (V1, V2, …, Vn)
Assignment vector, A = (V.size(),0)
Free space vector, F is an empty vector
Function first_fit(Vector V,  Vector A, Vector F){
        Set a flag new_bin = 0 to check should create a new bin to handle the item
        For I = 1 to n do
                For j = 1 to the size of F:
                        If V[i] less or equal to the F[j] then
                                F[j] is F[j] – V[i]
                                Assign the A[i] is j
                                Flag is 1
                                Break the for loop
                If new_bin not equal to 1 then
                        Create a new bin to load the V[i]
                        Push back the (1 – V[i]) into the F
                        Assign the A[i] is the size of F minus 1
}
```

Result of the First Fit

test_size	waste
4	0.216833
8	1.44903
16	0.863933
32	2.45528
64	3.82028
128	8.44046
256	12.3784
512	24.0153
1024	29.4882
2048	54.7603
4096	75.0707
8192	111.313
16384	193.9
32768	342.063
65536	588.757
131072	892.089
262144	1349.56
524288	2158.84



Analysis of the Result

From the above data and graph, we could find this is a linear function, and the function was y = 0.73x +-0.74. The slop was less the slope of the Next fit, so it created less wastes than the Next fit did.

Best Fit

Pseudo Code

Random items vector, V = (V1, V2, …, Vn)

Assignment vector, A = (V.size(),0)

Free space vector, F is an empty vector

Function best_fit(Vector V, Vector A, Vector F){

For i = 1 to n do

Scan all the existed bins to find the bin which has the tightest remain with the item.

If the V[i] larger than all the bins' remain then

Create a new bin to hold the item

Push back V[i] into the F

Assign A[i] is the new bin number.

Else then

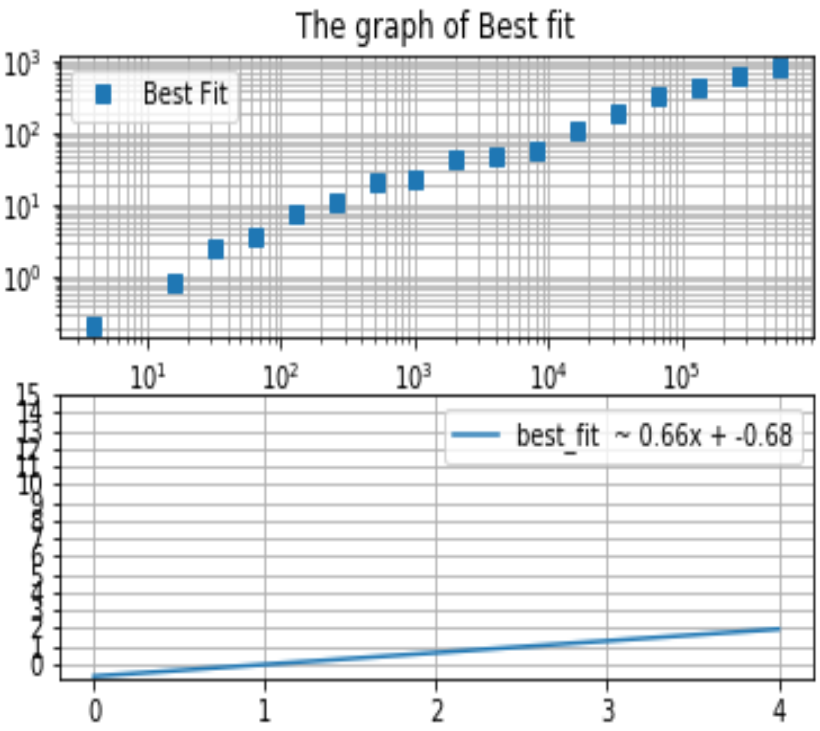
Add the V[i] to the bin which has the tightest remain with the item.

Assign A[i] is the bin number.

}

Result of the Best fit

test_size	waste
4	0.216833
16	0.863933
32	2.45528
64	3.82028
128	7.44046
256	11.3784
512	21.0153
1024	23.4882
2048	45.7603
4096	50.0707
8192	57.3133
16384	108.9
32768	200.063
65536	340.757
131072	455.089
262144	628.56
524288	829.836



Analysis of the Result

From the above data and graph, we could get this was a linear function, and the function was y = 0.66x + -0.68. From the data that I collected, I found best fit created wastes less than the First fit and Next fit did. It is a good way to do bin packing.

First Fit Decreasing

Pseudo Code

Random items vector, V = (V1, V2, …, Vn)

Assignment vector, A = (V.size(),0)

Free space vector, F is an empty vector

Function first_fit_decreasing(Vector V, Vector A, Vector F){

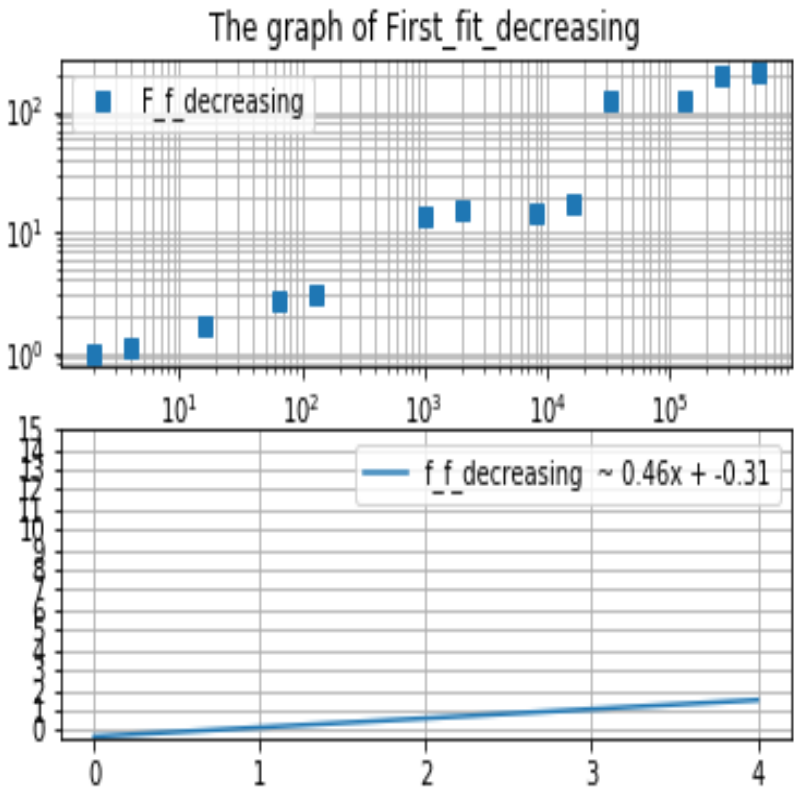
    Use the build in sort function to sort the V from largest to smallest.

    Then run the First fit algorithm

}

Result of the First fit decreasing

test_size	waste
2	0.987392
4	1.13061
16	1.69404
64	2.66768
128	2.99772
1024	13.5396
2048	15.1037
8192	14.7153
16384	17.2321
32768	125.418
131072	126.436
262144	196.719
524288	210.87



Analysis of result

From the above data and graph, we could get the function was a linear function, and the function was y = 0.46x +- 0.31. According to the data I collected, the first fit decreasing created less wastes than the Next fit and First fit and Best fit did.

Best Fit Decreasing

Pseudo Code

Random items vector, V = (V1, V2, …, Vn)

Assignment vector, A = (V.size(),0)

Free space vector, F is an empty vector

Function best_fit_decreasing(Vector V, Vector A, Vector F){

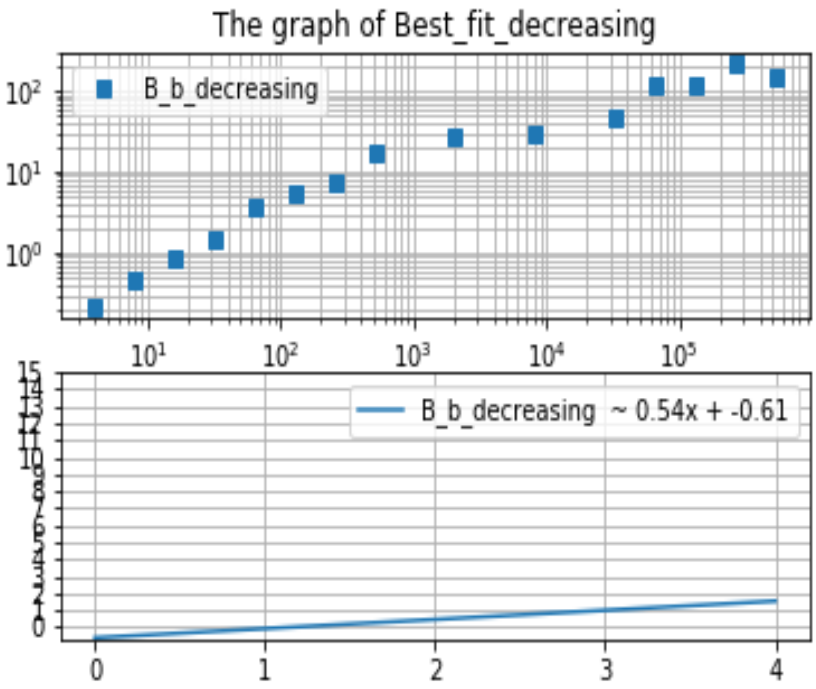
    Use the build in sort function to sort the V from largest to smallest.

    Then run the Best fit algorithm

}

Result of Best_fit_decresaing

test_size	waste
4	0.216833
8	0.449029
16	0.863933
32	1.45528
64	3.82028
128	5.44046
256	7.37839
512	17.0153
2048	27.7603
8192	29.3133
32768	45.0625
65536	115.757
131072	121.089
262144	217.037
524288	152.836



Analysis of Result

From the above data and graph, we could get this was a linear function, and the function was y = 0.54x +- 0.61. At the start of the algorithm, it would sort all items from largest to smallest. For example, After sorting the vector [0.7,0.8,0.9,0.6,0.4,0.3,0.2,0.1] will become [0.9,0.8,0.7,0.6,0.4,0.3,0.2,0.1]. All the item was larger than 0.5 would be stored in an independent bin, and all the items were less than 0.5 would be added to these bins. It was a good way to do the bin packing, and it would save many storages.

Conclusion

The Next fit would cause the most wastes.

The First fit would cause less wastes than the Next fit did. It is not as good as Best fit.

The Best fit created less waste than the First fit and Next fit, but it would take longer time than the First fit did for searching the tightest bin.

The qualities of First fit and Best fit were almost same. They created less waste than the above three. It is a good way to do bin packing.

Here is my assumption about why the qualities of First fit and Best were almost same. After the sorting, the First fit and Best fit will all start at the bigest item to the smallest item, so every item larger than the half of the bin will pack in an independent new bin. Because we using vector and push back to add the new remains to the vector, so the first bin was also the bin with the largest remain. For both algorithms, the first item which can fit into exist bins usually only smaller than the largest remain, so in this situation, the First fit has the same performance as the Best fit.

Citated Works

Wikipedia contributors. "Bin packing problem." *Wikipedia, The Free Encyclopedia*. Wikipedia, The Free Encyclopedia, 12 May. 2019. Web. 20 May. 2019.