Brandon Fowler
CSCD340
Homework 4

<u>Graphs</u>

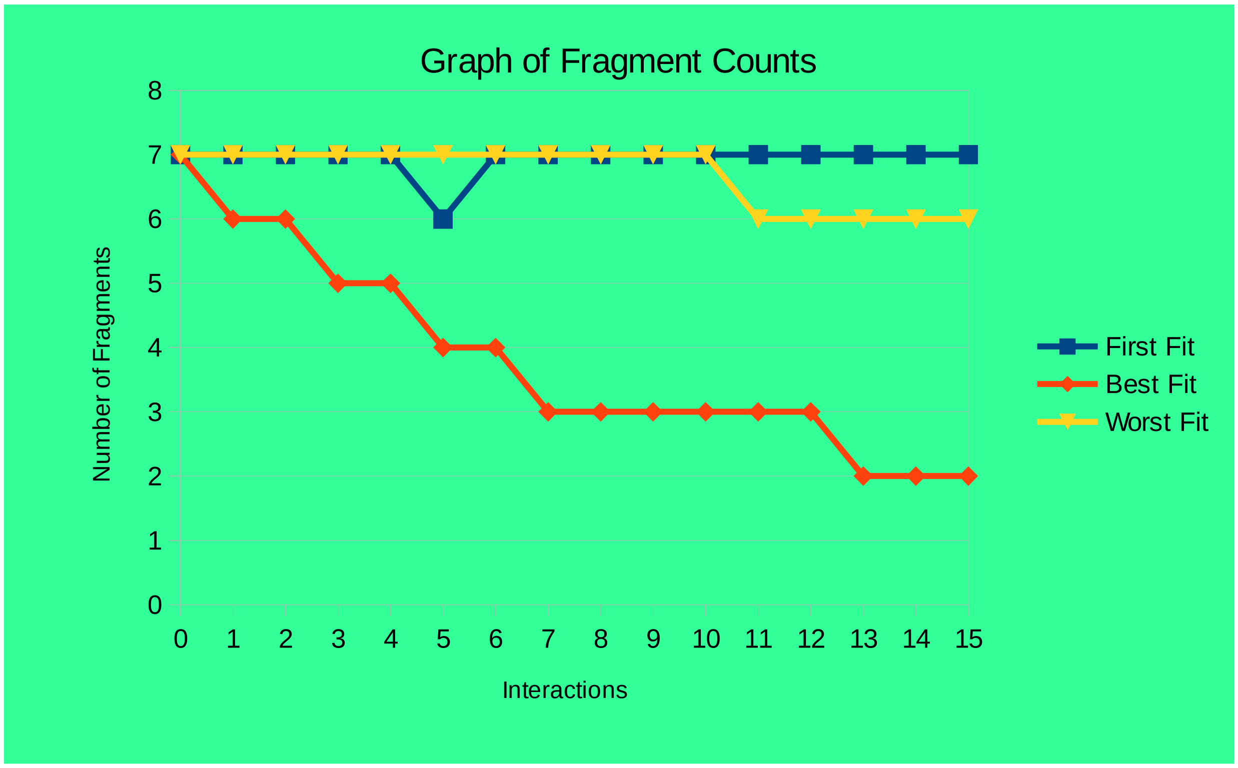
All charts below, are a graphs of statistics calculated for the same 15 interactions, after construction of the starting memory map.

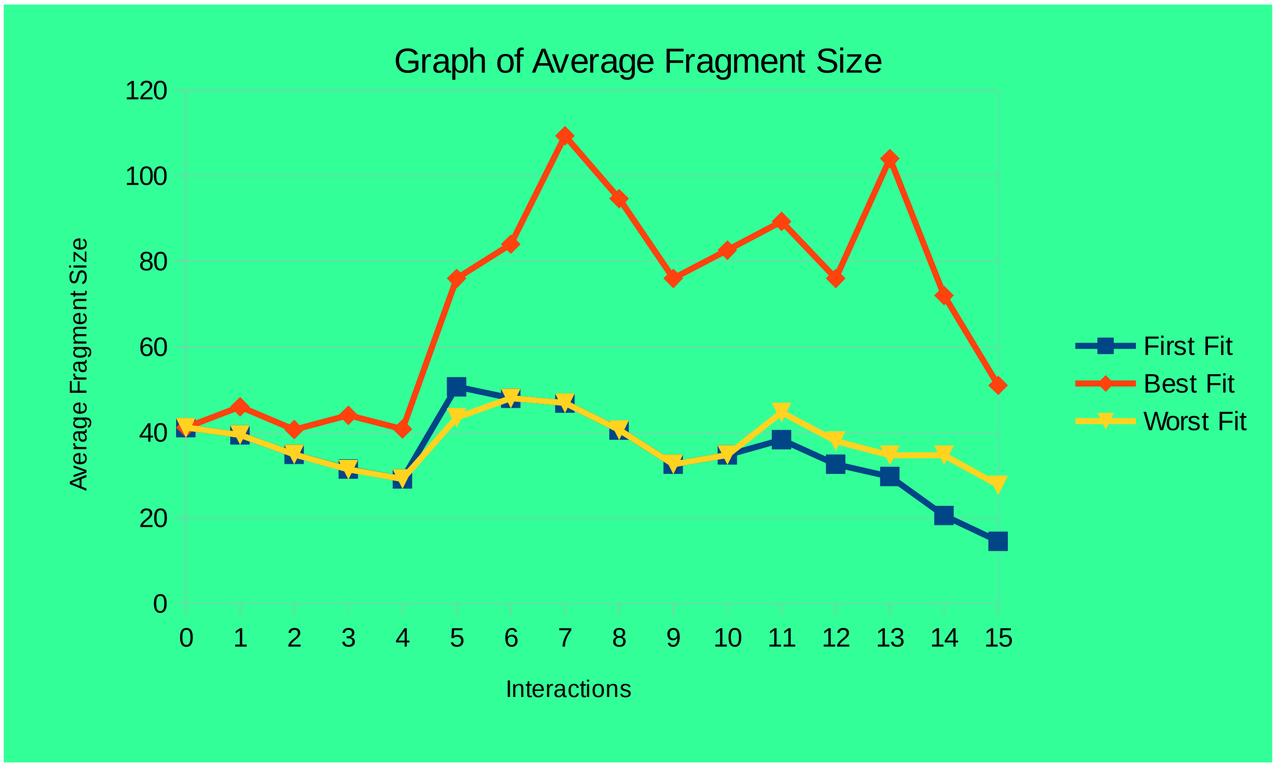
Starting memory map data:

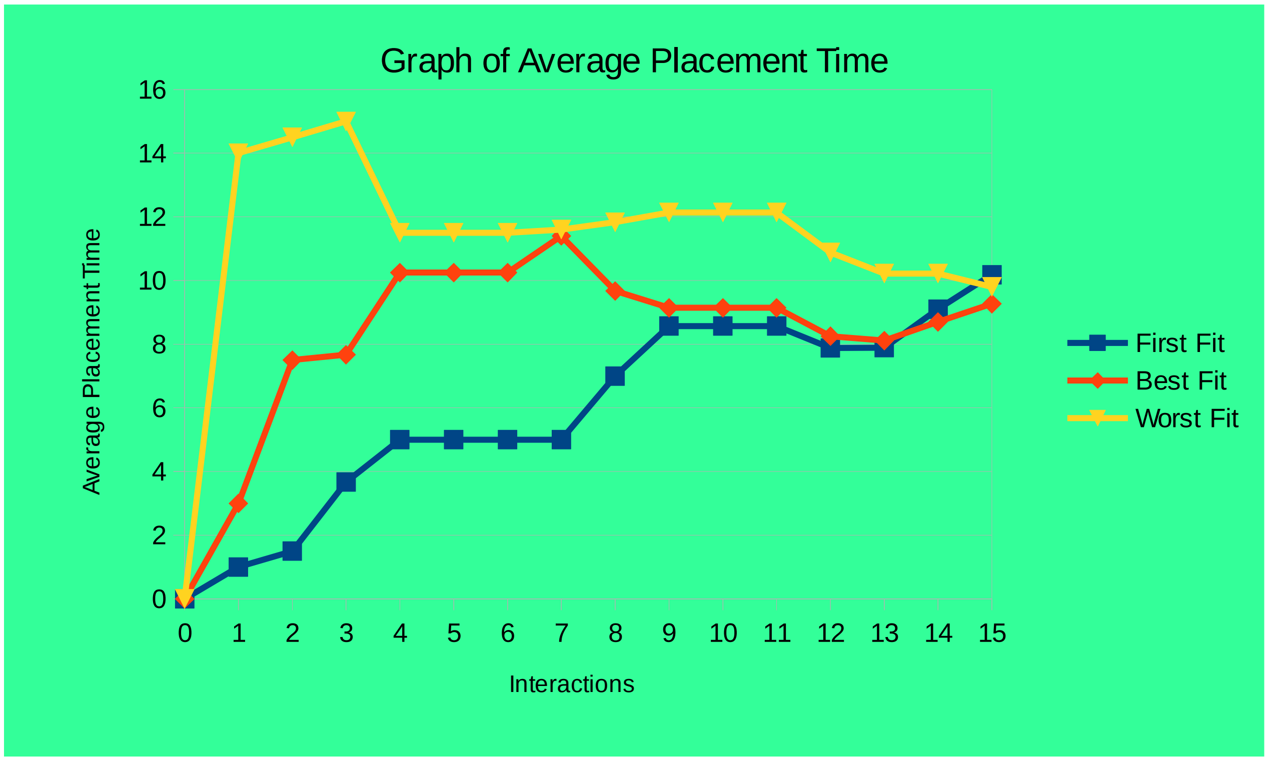
900 124
P 1 40 172
P 2 12 224
P 3 32 236
P 4 64 312
P 5 24 400
P 6 76 424
P 7 64 500
P 8 100 600
P 9 128 800
p 10 72 928

Interactions:

1. add process, ID 11, size 12
2. add process, ID 12, size 32
3. add process, ID 13, size 24
4. add process, ID 14, size 16
5. terminate process, ID 8
6. terminate process, ID 3
7. add process, ID 15, size 8
8. add process, ID 16, size 44
9. add process, ID 17, size 56
10. terminate process, ID 5
11. terminate process, ID 1
12. add process, ID 18, size 40
13. add process, ID 19, size 20
14. add process, ID 20, size 64
15. add process, ID 21, size 42




Graph of Average Fragment Size
Average Fragment Size
120
100
80
60
40
20
0
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Interactions
First Fit
Best Fit
Worst Fit


Graph of Average Placement Time
Average Placement Time
16
14
12
10
8
6
4
2
0
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Interactions
First Fit
Best Fit
Worst Fit

Analysis

This analysis covers the algorithms of First Fit, Best Fit, and Worst Fit in relation to the above test, resulting data, and graphs. Each algorithm started with the same memory map, then was applied with the same 15 commands. The total memory space was 900, and the starting point was 124, giving an end point of 1024. The initial memory map started with 10 processes of varying starting addresses and sizes, and 7 holes of varying sizes.

As shown in the above graph, the number of fragments for the Fist Fit algorithm and the Worst Fit algorithm, stay very close to each other through most of the test; and only diverge slightly when they do differ. Whereas, the number of fragments in the Best Fit algorithm consistently grows less, and ends much lower in fragment count that the other two algorithms. I expected to see several several small/unusable fragments resulting from the Best Fit algorithm, which I believe is still the usual case. However, based on this data, the best fit algorithm did the best job of lowering the amount of fragments.

In the second graph, we see the average fragment size comparison for each algorithm during the test. Again, the First Fit and Worst Fit algorithms remain very close to one another, with the First Fit algorithm averaging just barely less than the Worst Fit algorithm. The Best Fit algorithm ranges up and down drastically in average fragment size as compared to the other two, but always remains larger. Again I did not expect this from the Best Fit algorithm, and had initially assumed the average fragment size would be less due to many small fragments. However, in this example I can see that most of the processes were small in size compared to a very large hole towards the end of the map. That hole was never filled and consistently became larger in the Best Fit algorithm, because new processes fit better in other holes and terminating processes expanded the hole. In addition several of the processes fit perfectly into some holes. This caused the average fragment size to remain much higher. In this example the First Fit algorithm had the lowest average fragment size, due to many smaller fragments in lower memory. However, it was followed very closely by the Worst Fit algorithm, and I believe long term that they would remain similar. However, it is worth noting that since the Worst Fit algorithm has a more even size distribution of holes, and eliminates the larger holes seen in the other algorithms; it is the most likely to have insufficient space if a large process is entered.

In the last graph, we see that in this example the average placement times vary greatly from the start, then draw closer together. I expect normally to see the Best Fit and Worst Fit algorithms to be very similar in placement time. However, since in this example the memory map started with a large hole in higher memory; it makes sense that the Worst Fit algorithm should begin with a higher average placement time, since it had to search farther. As expected, the First Fit algorithm generally had the lowest average placement time, since it simply uses the first space that is big enough for the process.

In general, I would not say that one algorithm is better than another, as they all have their uses. First Fit is best when speed of placement is the main concern. Worst Fit is best if processes are not too large, and the benefit of a more even free memory distribution is worth the speed loss. Lastly Best Fit can be useful if process vary greatly in size, and the need is greater to conserve free memory for larger processes than a need for greater speed.