The University of Texas at Arlington

# Assignment #3 – Malloc

Joshua Catalan

CSE 3320-003

Trevor Bakker

14 April 2023

## Executive Summary

In this assignment I was tasked to create my own implementation of malloc, free, calloc, and realloc. I also had to implement three additional heap management strategies such as Next Fit, First Fit (Done for us), Best Fit, and Worst Fit. Within my malloc function, I had to account for free space in a block by splitting the block if it had enough space remaining for another allocation. With my free function, I had to account for block coalescing. If the next block or previous block were free, then I would have to combine them with the current block that's being freed. Calloc and Realloc are basically just malloc calls with a slight difference. Calloc allocates a block of memory and initializes it to zero. Realloc resizes a previously allocated block of memory. To traverse through the heap, a linked list must be used to easily connect each block and perform pointer arithmetic if need be.

## Description of the Algorithms Implemented

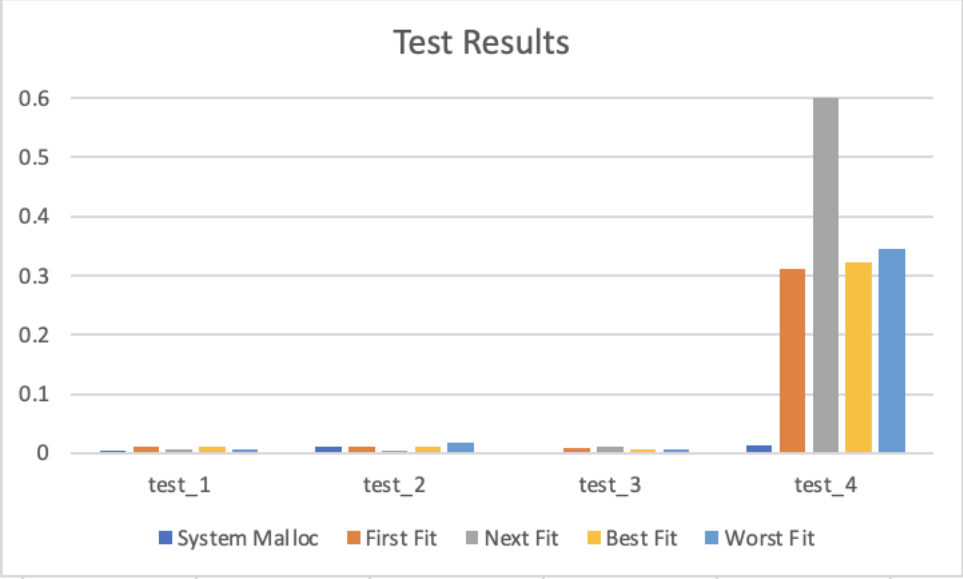
I had to implement Next Fit, Best Fit, and Worst fit in my malloc program. First Fit was done by the professor and if the current block was not NULL, not free, and the block has a size that is less than the requested size then the list will keep traversing until it found an appropriate block to allocate. Best Fit traverses the same way as First Fit but has a few differences in choosing its block. It will traverse the linked list until it found a block that was no NULL and had a size greater than the maximum integer possible. Worst Fit behaves the exact same way as Best fit but the only difference is that instead of comparing with the maximum Integer, it compares with the minimum integer. Next fit will keep iterating forward until it reaches then end of the list, then would start back from the top and end where it previously started. It also traverses the list similarly, but the only difference is that it wraps back around the list.

- First Fit: searches the available memory blocks starting from the beginning of the heap and allocates the first block that is large enough to hold the requested memory.
- Next Fit: searches the available memory blocks in a circular manner, starting from the location of the last allocation and wrapping around to the beginning of the pool if necessary.
- Best Fit: searches the available memory blocks and allocates the block that is closest in size to the requested memory.
- Worst Fit: searches the available memory blocks and allocates the block that is furthest in size to the requested memory.

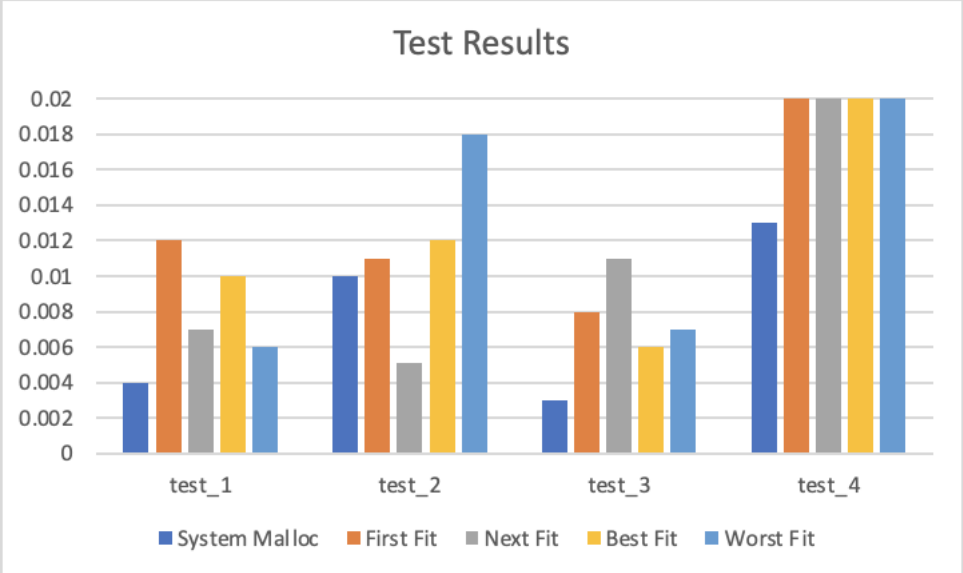
## Test Implementation

| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| 1 | | System Malloc | First Fit | Next Fit | Best Fit | Worst Fit |
| 2 | test_1 | 0.004 | 0.012 | 0.007 | 0.01 | 0.006 |
| 3 | test_2 | 0.01 | 0.011 | 0.005 | 0.012 | 0.018 |
| 4 | test_3 | 0.003 | 0.008 | 0.011 | 0.006 | 0.007 |
| 5 | test_4 | 0.013 | 0.311 | 0.608 | 0.322 | 0.346 |

Above is a table with the system malloc call and the 4 heap management strategies performance results on each of the tests I created.




Y-axis shows performance in seconds and the X-axis shows the respected test.




Same test results just adjusted for test_1, test_2, and test_3 to see the data clearer.

## Interpretation of Results

- Test_1: The system malloc call performed the fastest while First Fit performed the slowest.
- Test_2: Next Fit performed the fastest while Worst fit performed the slowest.

- Test_3: The system malloc call performed the fastest while Next Fit performed the slowest.
- Test_4: The system malloc call performed the fastest while Next Fit performed the slowest.

These are the average placings for the following: system malloc (1.25), First fit (3.75), Next Fit (3.0), Best fit (3.25), Worst Fit (3.5). As you can see on average the system malloc call was the fastest followed by Best fit, Next Fit and Worst Fit were tied, and lastly First Fit.

## Conclusion

In my tests, the system malloc call ran the fastest (1st), Best Fit ran 2nd fastest, Next Fit and Worst Fit tied for 3rd, and lastly First Fit was the slowest at 4th place. One anomaly that I found was that all 4 heap management strategies gave me the same statistics at the programs exit. Upon further research, I found that First Fit would have more heap fragmentation than the rest and should have more splits than Best Fit but less splits than Worst Fit. Also found that First Fit should generally run faster than Best and Worst fit, but in my results First Fit was the slowest heap management strategy. Best and Worst fit help reduce heap fragmentation but are typically slower since they must iterate through the entire linked list to find the winning block. Worst Fit typically has fewer splits and less heap growth. Best Fit typically has less splits and heap growth than First Fit but is comparable to Next Fit. Lastly, Next Fit is typically faster than First Fit and Best Fit but slower than Worst Fit. The number of splits and heap growth is comparable to First Fit, but can lead to more heap fragmentation due to leaving small gaps between the allocated blocks.