A4: Malloc Replacement Assignment

Tailang (Terry) Cao u1480633

Testing Methodology

I put all my test in the main.cpp file, with the 'test' function testing the constructor, destructor, insert, delete and all the other functions, and the 'benchmarkTest' recording the runtime of allocating and deallocating comparing with the built in malloc. The main function calls the two test functions and print out the result.

Runtime Performance

My code test the average of runtime over 1000 iterations as the result. In each iteration, I record the runtime of allocating 1000 pointers and freeing the memory. The result is shown below in nanosecond.

Average MY ALLOCATED/MALLOC time in nanoseconds was: 991810

Average REAL malloc time in nanoseconds was: 19286

Average MY DEALLOCATE/FREE time in nanoseconds was: 734522

Average REAL free time in nanoseconds was: 27891

Analysis

As is shown in the result, MyMalloc has a significantly worse performance comparing with the built in Malloc. This is primarily due to their underlying memory allocation strategies. MyMalloc relies on *mmap* and *munmap* system calls for memory allocation and deallocation, which are inherently slower compared to the strategies employed by *malloc* and *free*. Also, the built-in *malloc* implementation often incorporates optimizations to efficiently handle various allocation patterns, thus resulting in higher performance comparing to MyMalloc, which rounds up allocations to the nearest page size for each operation.

Based on the analysis above, we can increase the performance of MyMalloc by reducing the time of using system calls. This could be done by asking for larger blocks of memory in advance and divide them into smaller blocks when needed instead of calling mmap each time.