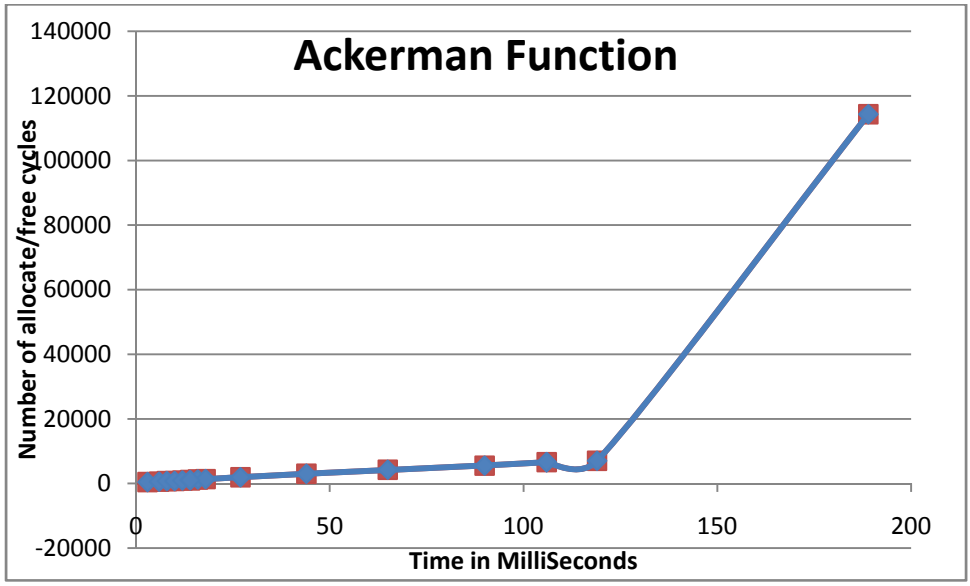
Analysis by Andrew Johnson



This clearly shows an exponential relationship between the number of Allocate cycles and the time taken to process the table.

The most likely cause is because of the Fibonacci function in the allocator. Every allocation requires many Fibonacci calculations.

Since this is the main bottleneck in the program. Increase the speed required to calculate the different numbers would increase the speed dramatically. I would pre-calculate the numbers which would require more memory, but it would make the function linear.

I have used a very poor implementation because I calculate the Fibonacci number for each allocation and de-allocation several times. To increase the functions speed I would keep the index and the Fibonacci number in the struct for each node. It would require more memory, 8 bits per node however, the de-allocation could be done without calculating the Fibonacci sequence at all.

To compile the program

GCC memtest.c my_allocator.h my_allocator.c ackerman.c ackerman.h

To run the program

./a.out

I used the atexit function to make sure the memory was freed

I used the getopt function to get the arguments from command line