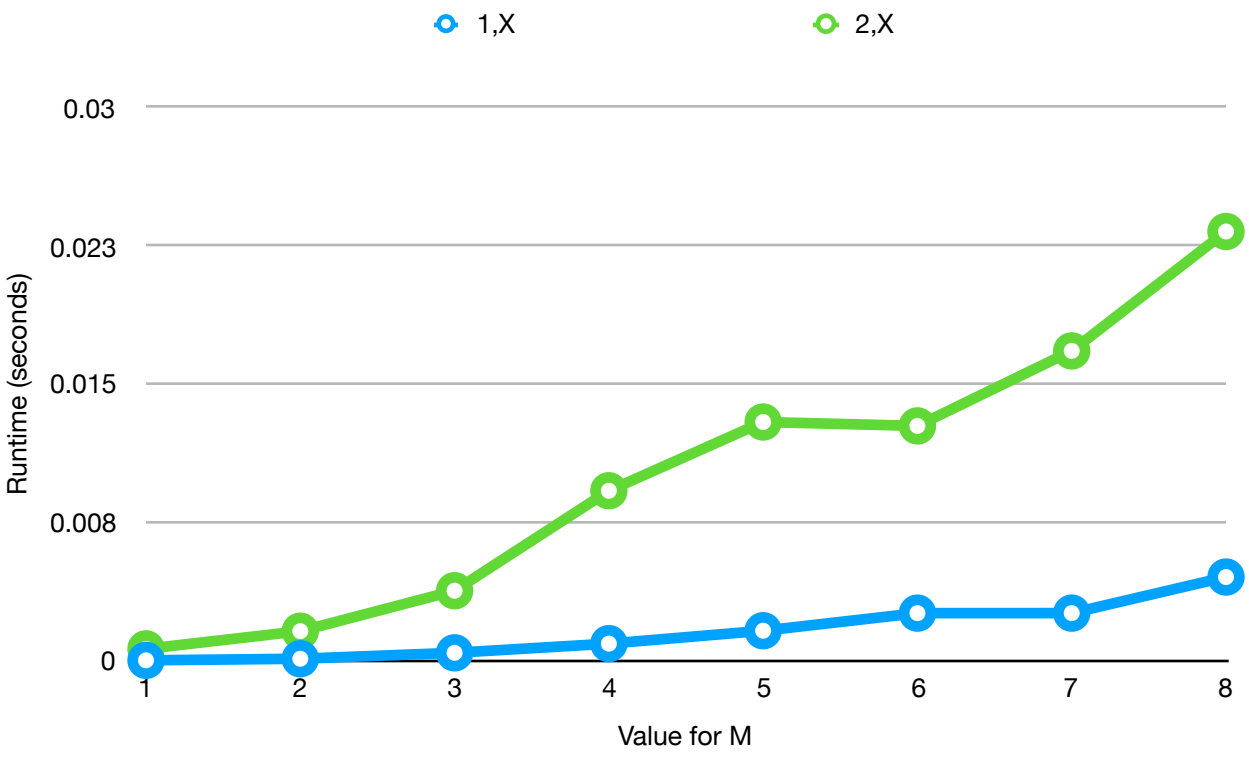
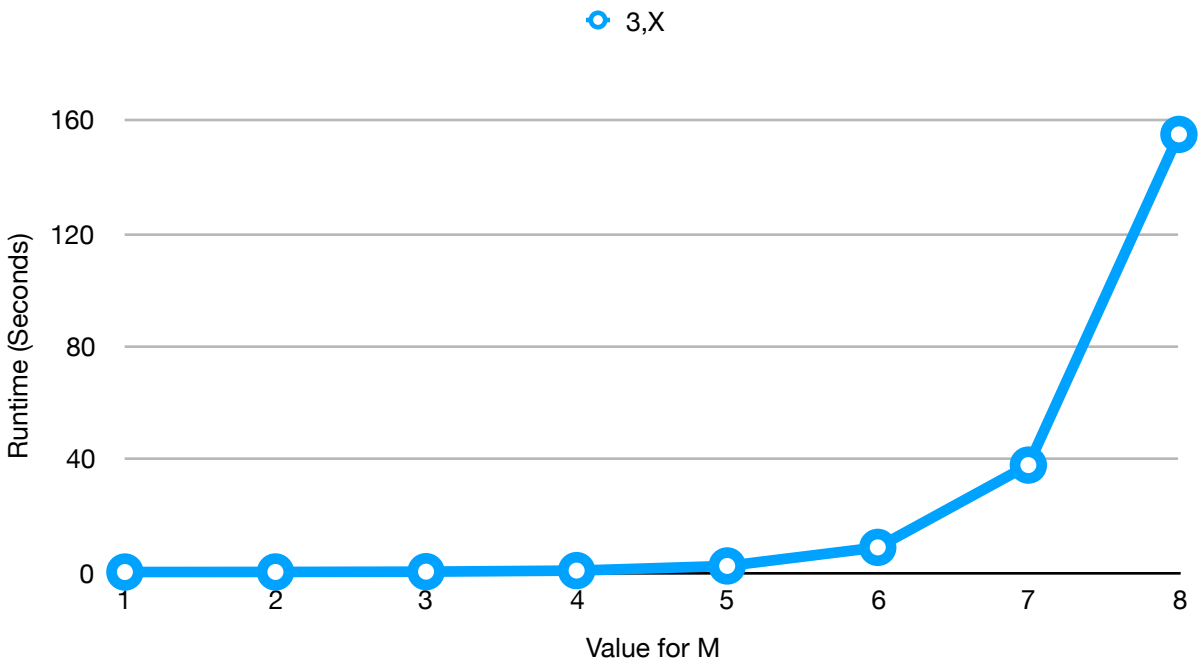
Introduction:

In this PA, we implement a memory manager using the Buddy System algorithm. In this algorithm, memory is represented as a max-heap, in which memory is recursively split/merged in terms of 2^x such that a block just large enough for the users request is allocated to the user.

Runtime Analysis:





Ackerman runtime increased in direct relation with the value of N. Runtime growth rate was exponential with n-values > 3. This growth rate of runtime is directly related to the number of recursive calls within Ackerman, which rises with 0(n^2) time complexity. The time complexity of the Buddy system allocator is log(N), where N is the number of block sizes. This is due to the fact that the worst case number of merge() and split() calls is equal to the number of levels of the max-heap, which has a logarithmic relationship with the number of nodes. One major bottleneck is the use of LinkedLists inside each FreeList element, each of which has a 0(N) search complexity. Integrating a hash table will allow for near constant O(1) lookup times when searching for a particular blockheads to remove. This can also be expanded to FreeList, to minimize iteration of the FreeList by conducting a hash table lookup for the FreeList indices that have available memory blocks to allocate.