Henry Daniels-Koch and Jiun Kim
October 14 2016

Algorithms: Lab 4

We explore two different sorting mechanisms for both integers and doubles from 0 to 5 million. We test both Quicksort and Heapsort algorithms and evaluate their runtimes for various array lengths N. To generate the array of numbers, we use a random number generator that uses the time as the seed and generates a random number from 0 to 5 million. We use either Quicksort or Heapsort to sort the list. We then check whether the list is sorted to confirm our algorithms work. We use a 4GHz Intel Core i7 with 16 GB iMac (computer in 224).

We have plotted run times for Quicksort and Heapsort vs array size in the plot attached. In general, we observe Quicksort sorting between 2 and 6 times faster than Heapsort for doubles. We stress that this data is specific to the range of our random integers. With a range of 0 to 5 million, we expect that Quicksort's speed will not be as impressive as the size of the array since Quicksort does not partition effectively when there are many duplicates as might be the case with an array of 1 billion. HOWEVER, we instead see that the ratio of runtimes of Quicksort to Heapsort peaks when $N = 10million$ where QuickSort is 5.2 times as fast. Perhaps at $N = 100million$, the duplicates in Quicksort slow it down forcing it to only be twice as fast. What is strange is that at $N = 1billion$, QuickSort becomes 3 times as fast as Heapsort. Maybe memory allocation is driving this strange behavior.