

As we can see, the running time of mergeSort is much quicker than bubbleSort. When we increase the input size from 1000 to 4000 and to 4551 (the entire dataset), the running time of mergeSort increase only about 0.001 seconds whereas the running time of bubbleSort increases significantly as shown in the graph. This probably has something to do with the different algorithm of two methods and the Big O.

The Big O of worst-case scenario of bubbleSort is O(n^2), when array is reverse sorted. This requires a lot of computation and comparison, sometime not necessary. It only sorted the adjacent pairs and repeat this over and over. By contrast, the Big O of worst-case scenario of mergeSort is O(nLogn). This is because merge sort split the list into half (which is O(logn)), until the entire list become a single unit and then sort it and merge back(which is O(n)).

O(nLogn) is not faster than O(n) when input size is small, but when input size increases, as shown in the graph, n square will be huge. nLogn will increase in a relative slower pace with increasing input size and keep a very short running time.