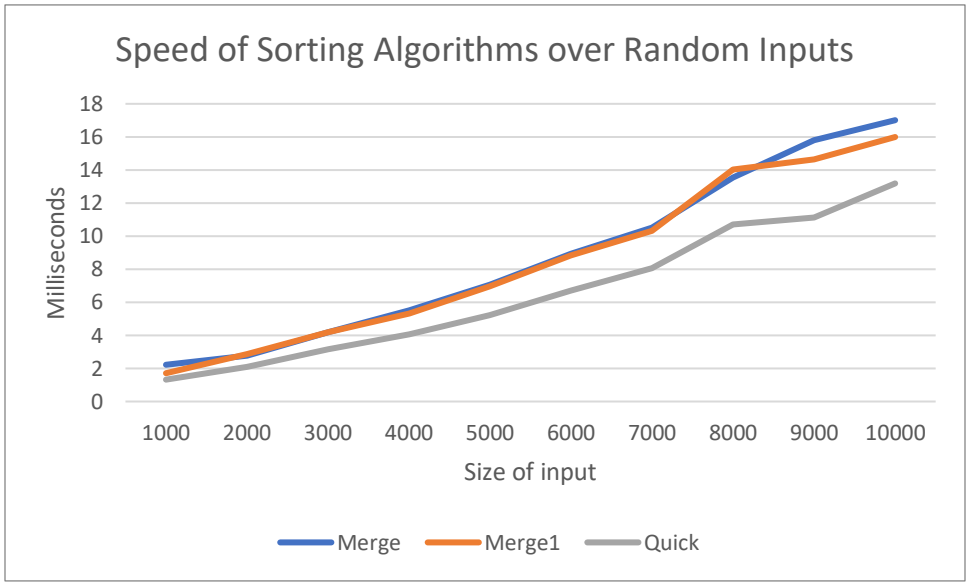
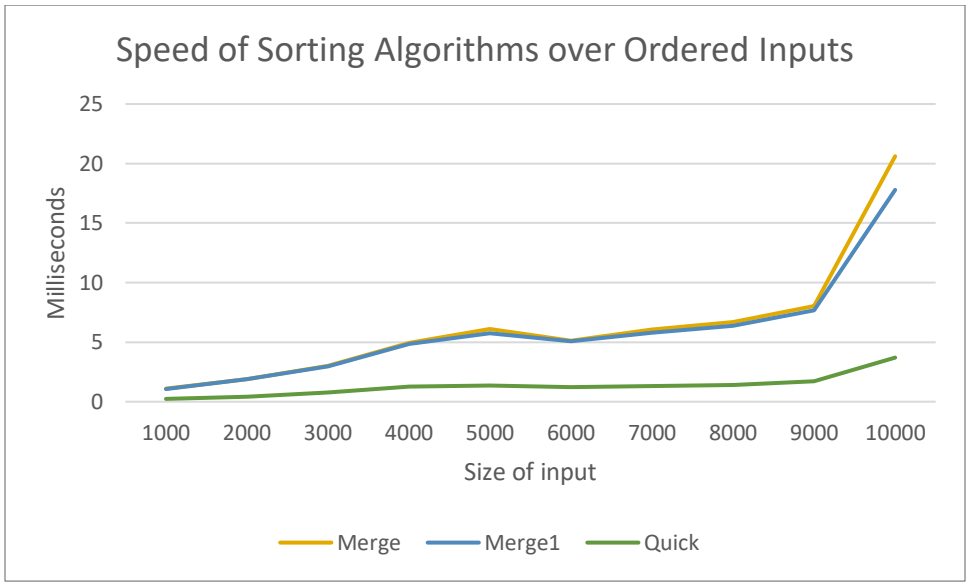
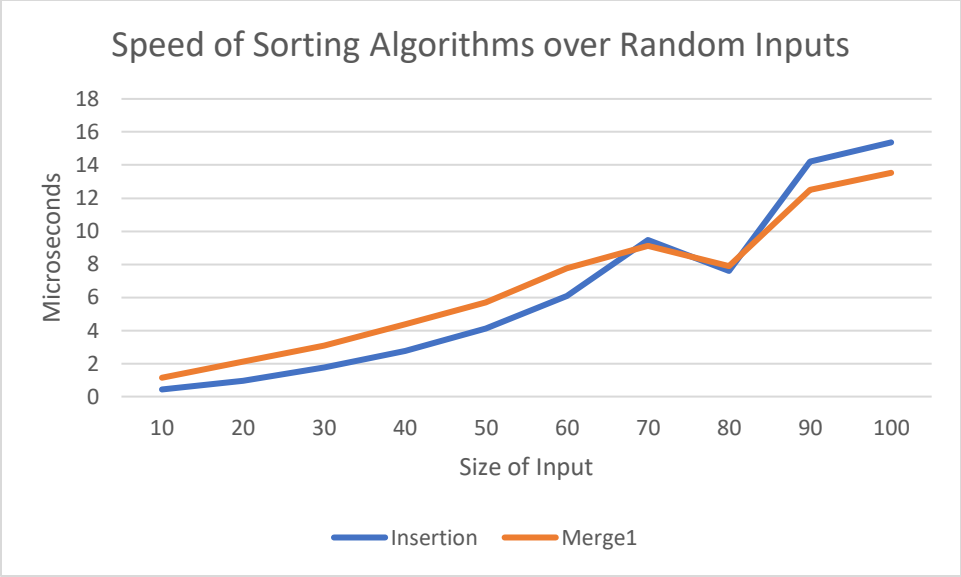
1. **Improved Merges**





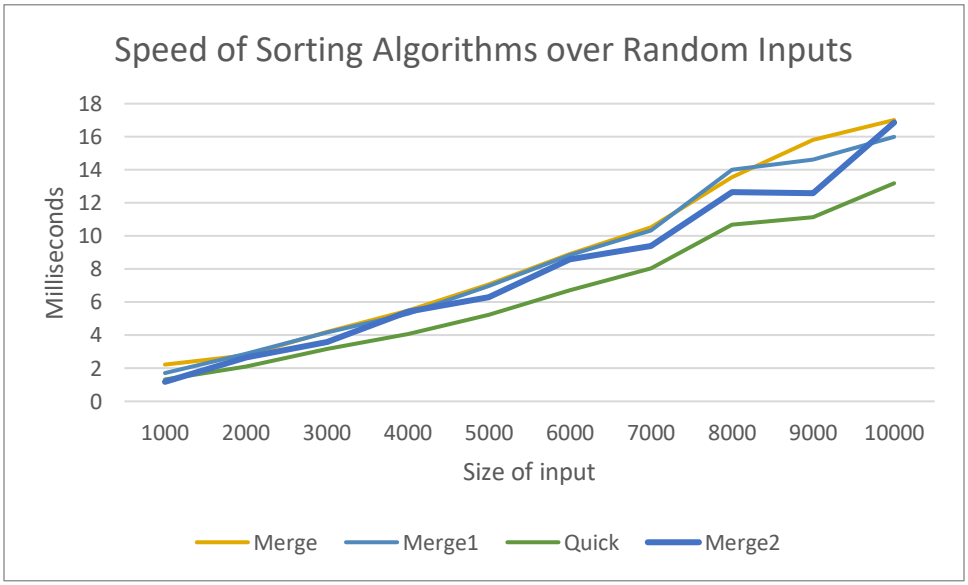
As we can see, quick sort is faster by a wide margin over this set of inputs, and merge1 is noticeably faster than merge only at larger inputs.

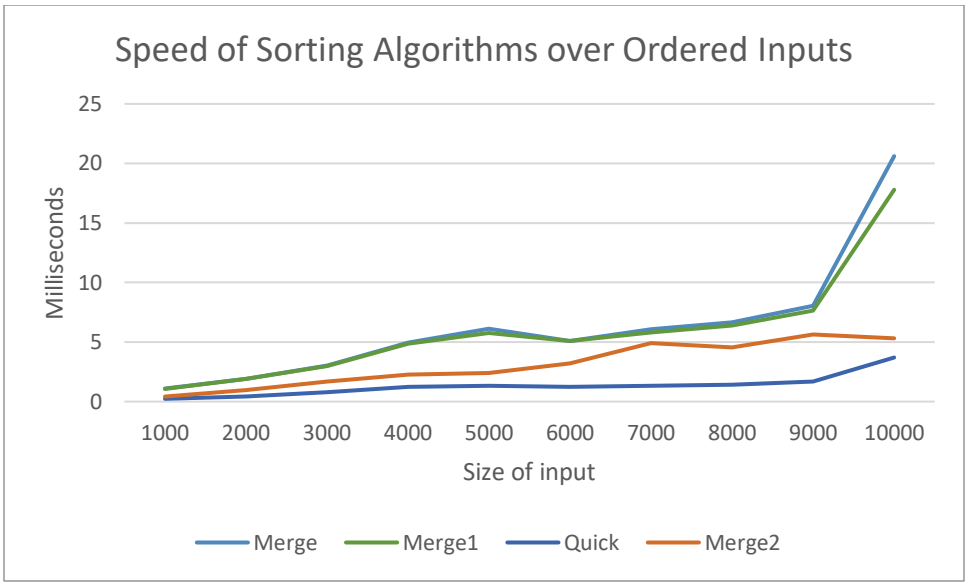
2. **Threshold selection**



Insertion sort is faster over small inputs, but the merge1 algorithm overtakes it at around the n≈85 input size. Thus, the merge2 code switches to insertion sort for subarrays at sizes of 85 and below.

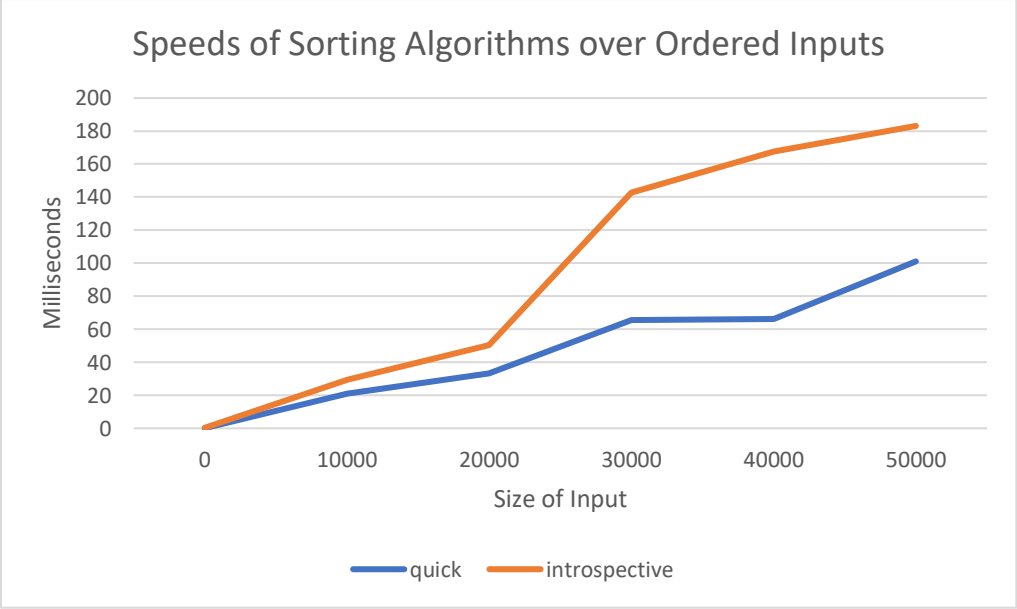
3. Switching Strategies





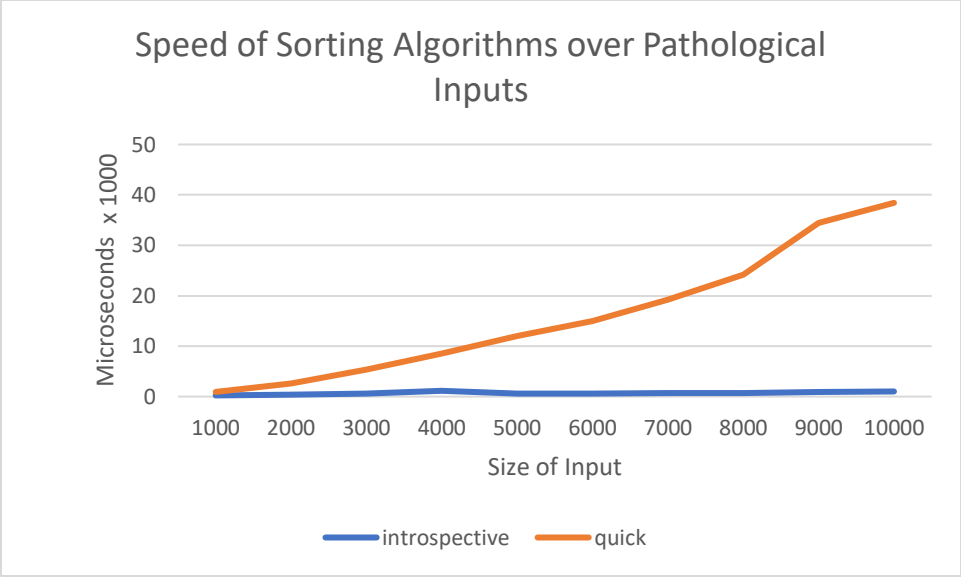
Adding Merge2 to the data shows that it is consistently faster than merge and merge1 at sorting, and even does very well at large, ordered inputs as well. However, it is still slower than quicksort over this input set.

4. **Introspective (Random Inputs)**



Over random inputs, quicksort is marginally faster than my current implementation of introspective sort, especially at very large input sizes. This is likely due to imperfections in my implementation, but an ideal introspective sort should be as fast or faster over all input sizes.

5. **Introspective (Pathological Inputs)**



Introspective sort is shown to be astronomically faster than quick sort over pathological inputs. The set of input sizes differs from the previous chart, as the previous input sizes threw a stack overflow error.