Andrew Smith, Dakota Donner, and Miguel Lopez

Week 6 Lab

Random Arrays

Merge sort and Quicksort both had drastically less comparisons than the “slow” sorting algorithms. Merge sort featured the lowest number of comparisons, with slightly more moves than comparisons, while Quicksort had slightly more comparisons and significantly more moves than Merge sort. Both “fast” algorithms, however, had significantly more moves than Selection sort. Even with this in mind, Merge Sort was likely the most efficient, followed by Quicksort, Selection sort, Insertion sort, and Bubble sort.

Already Sorted

Insertion Sort and Bubble Sort featured the lowest number of comparisons, but Merge Sort was not significantly higher. Quicksort was almost double the number of comparisons, and Selection Sort had by far the highest number of comparisons. No “slow” algorithm featured any moves, but both “fast” algorithms did, with Quicksort performing more than Merge Sort. Bubble and Insertion Sort seem to be the best for this type of array, followed by Merge Sort, Quick Sort, and dead last by Selection Sort.

Nearly Sorted

Insertion Sort performed the fewest comparisons and tied with Bubble sort for fewest moves. Merge Sort performed second in comparisons, but both “fast” algorithms had vastly more moves than the “slow” algorithms. Insertion Sort performed the best here, followed by Bubble Sort, Merge Sort, Quicksort, and Selection Sort.

Reverse Sorted

Merge Sort performed by far the fewest number of comparisons, while Selection Sort performed the fewest moves. Quicksort performed better than the other “slow” algorithms but performed significantly worse than Merge Sort. Merge sort seems to have performed the best, followed by Selection Sort, Quicksort, Bubble Sort and Insertion Sort.

Many Duplicates

Merge Sort performed the fewest comparisons, while Selection Sort performed the fewest moves. While Quicksort performed more moves than Selection Sort, it performed vastly fewer comparisons. Merge Sort seems to have performed the best, followed by Quick Sort, Selection Sort, Insertion Sort, and Bubble Sort.

Merge Sort seemed to have performed the best overall, with other algorithms performing better in certain circumstances. Namely, the “fast” or “efficient” algorithms get significantly less efficient if the array is already sorted or mostly sorted.