Sort Report

Ryan Millares

December 2019

1 The report

The time differences in the algorithms were pretty much as I expected; quick sort tended to be the fastest of them all on average, followed by merge sort, then bubble, selection, and insertion sorts sort of mixing around in their order. In a few occasions, quick sort would actually take longer, possibly as a result of my CPU and whatever processes it was happening to run at the time. When it came to picking certain algorithms over others, merge sort's trade-off was the extra memory requirement, and quick sort's trade-off was the complicated implementation. Selection and insertion work best if time isn't too much of an issue, and bubble sort is absolutely atrocious with larger data sets. I'm not sure what to comment on choice of programming language, given that we had to do it in C++, but I think the main impact it had was the specific implementation, and what library we used to store the times. This sort of empirical analysis has the shortcomings in that it requires real-world data, so to make it as accurate as possible would require testing the sorting algorithms against many quantities of data inputs, which would be heavily time-consuming and would have a heavy toll on CPU performance and resource-usage.