Karl Kirch
Assignment 1 CS5413

Performance analysis of sorting

A naive analysis of the problem in the assignment would lead one to think that choosing an optimal O(nlgn) sorting algorithm would give us the best performance as we get larger numbers of records. However, in this case we're adding each new record to the already sorted list. This would put us dangerously close to the worst case {O(n^2)} of a standard quicksort per pass (ie record) but alternatively puts us very close to the best case for a simple insertion sort {O(n)} pass. For my implementation I tried a quicksort implementation which ended up being very close to O(n^2) per pass as expected and then refactored to use a simple insertion sort. After using the insertion sort, I ended up with an O(n) per pass. In the end because there are n passes for each of the 6 sorts, the total time complexity ends up being around 6n(n+1)/2 which ends up being O(n^2) overall. This is still better than even the best case for a quicksort which would end up being O(n^2 * lgn).

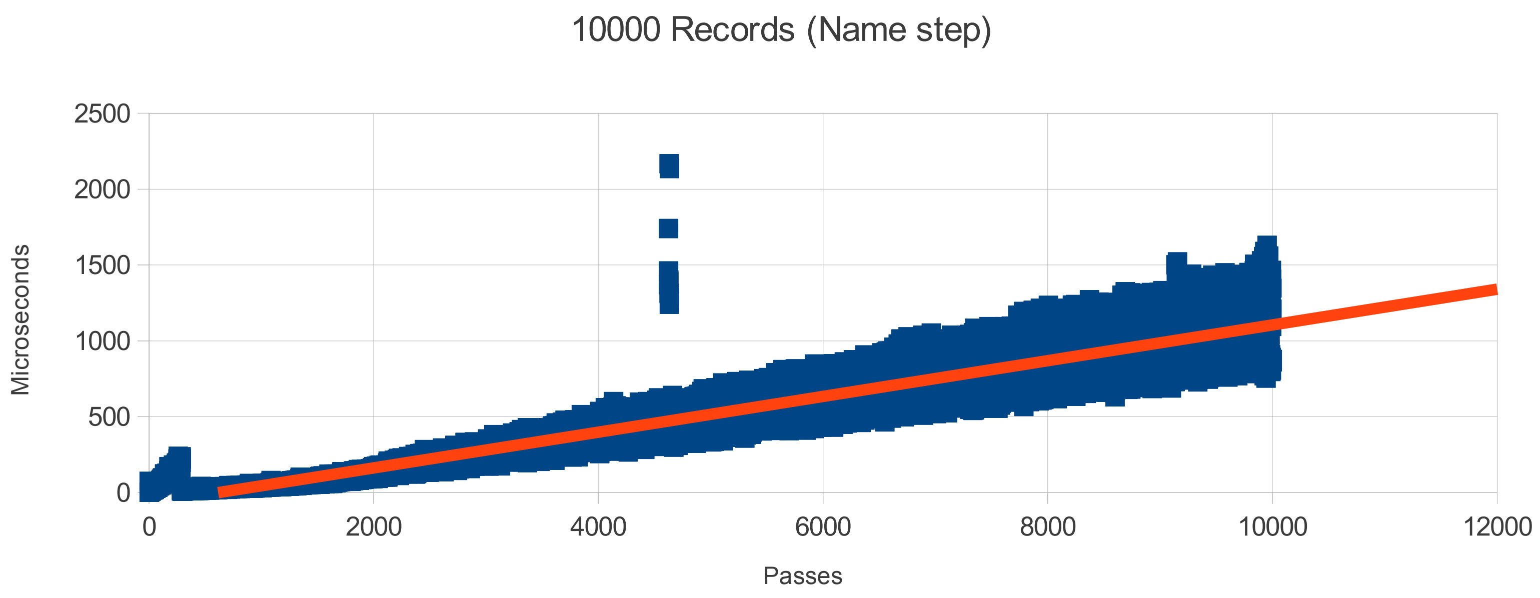
The graphs on the next pages illustrate the difference between the insertion sort implementation and the quicksort implementation. The insertion sort implementation clearly follows a linear path while the quicksort implementation follows an O(nlgn) path. Also of note was the difference in the actual running times of the algorithms. The insertion sort implementation ultimately taking around 1,600 microseconds for the 10,000th pass wheres the quicksort implementation taking around 30,000 microseconds for it's 10,000th pass.

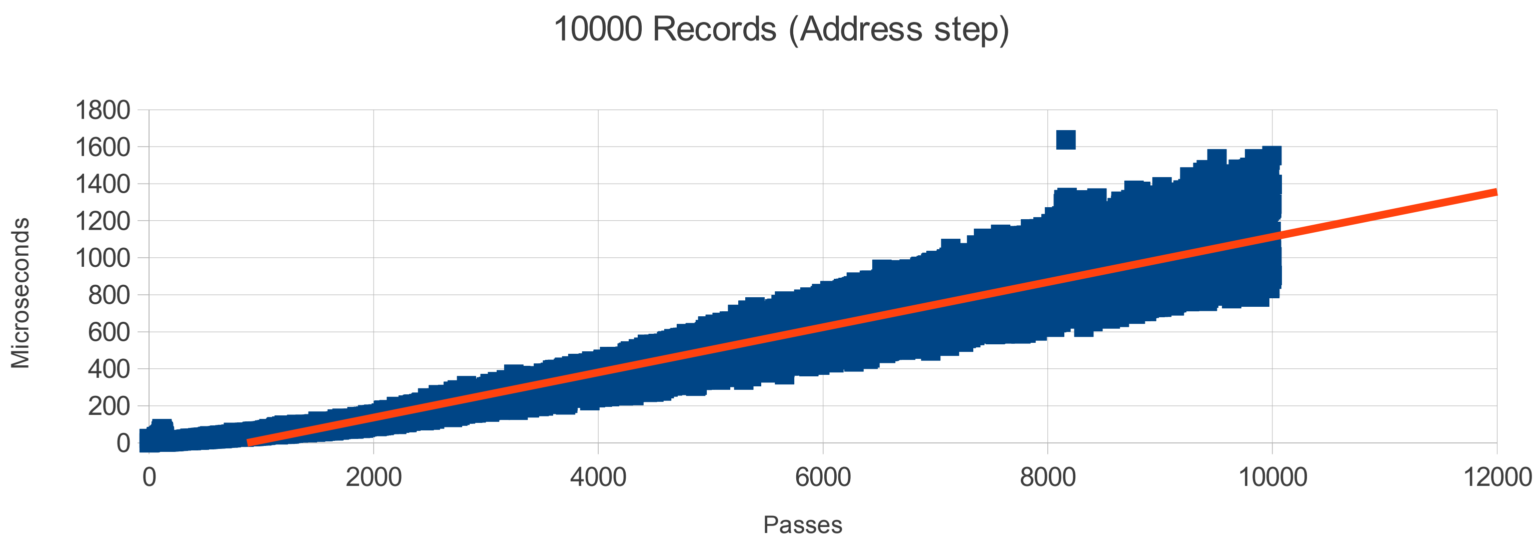
Ultimately (minus some outliers and some artifacts from running on the JVM) my implementations performed closely to their expected theoretical behaviors under the given datasets and constraints.

Though not explored, I could have modified the insertion sort to disregard the sorted portion of the list and only taken into account the single unsorted key, because of the specific constraints of having a completely sorted list that we are adding a single key to. This would have given that implementation an upper bound of Ω(n).

Karl Kirch
Assignment 1 CS5413

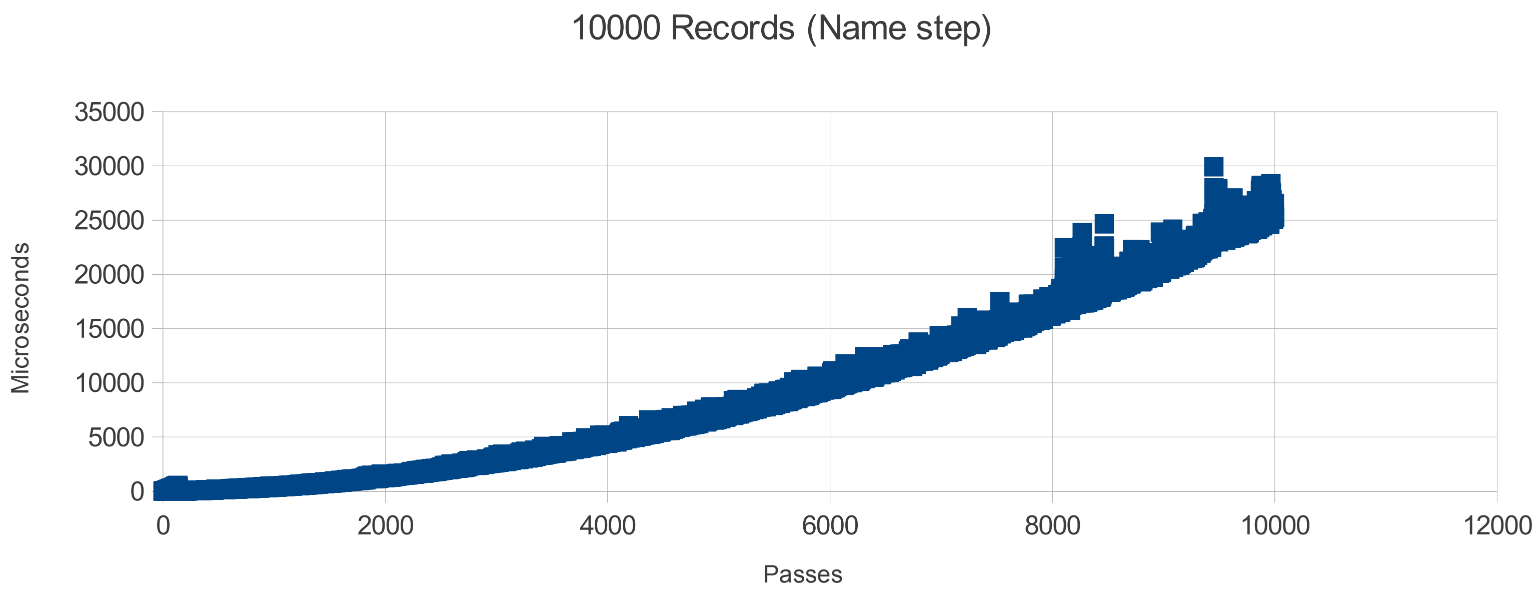
Graphs for per pass insertion sort timing

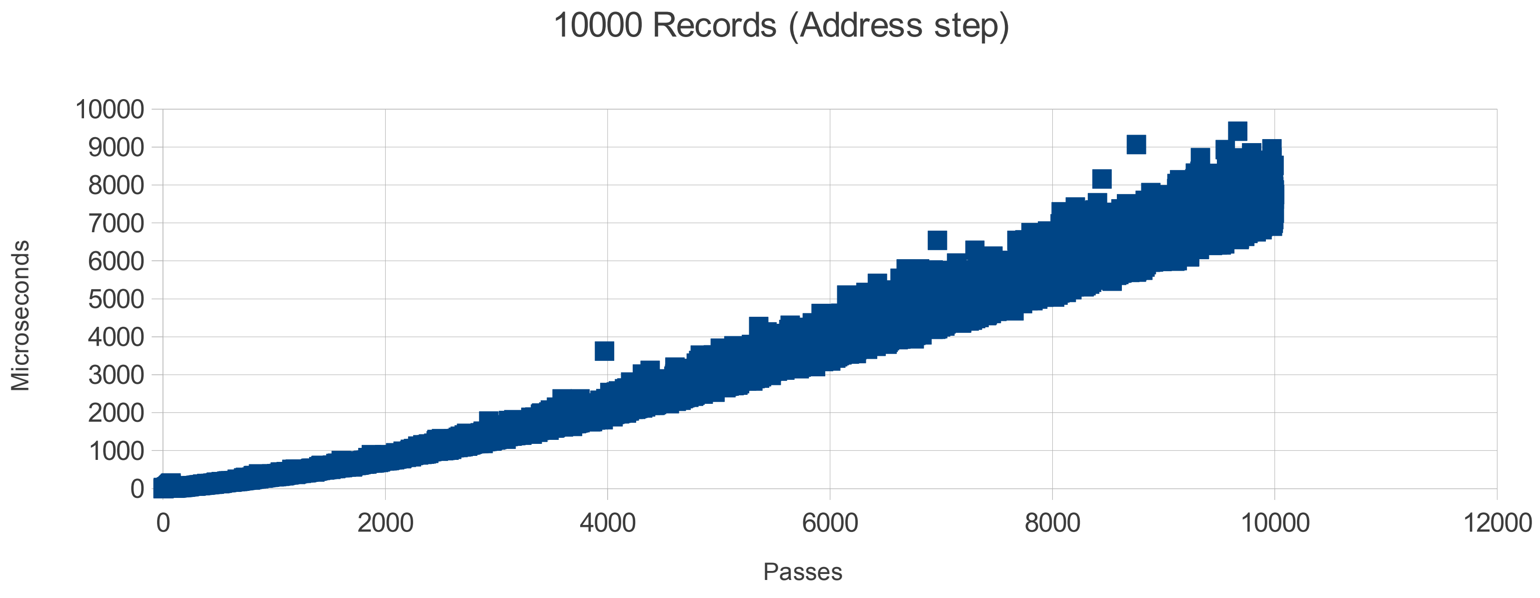




Karl Kirch
Assignment 1 CS5413

Graphs for per pass quicksort timing


10000 Records (Name step)
Microseconds
Passes


10000 Records (Address step)
Microseconds
Passes