Adam Schaible

CS4050

9/5/19

What is Quicksort and are some versions of Quicksort faster than others?

Quicksort is a procedure for sorting elements in an unsorted list into a list that is sorted from smallest to largest. (QuickSort) An example of elements includes numbers such as integers. (QuickSort) In Quicksort the unsorted list is divided into smaller lists where the elements in the smaller lists are sorted and are then merged back together in ascending order. (QuickSort) Some of the different versions of QuickSort includes not also the basic version of QuickSort, but also includes different strategies to partition the unsorted list into smaller lists. (QuickSort) The different QuickSort partition strategies also include the median of 3, Hoares, Lomato and Insertion Sort with a possibility of various region sizes. (QuickSort) Does the median of 3 make QuickSort run faster? Is there an optimal region size to first run QuickSort on followed by Insertion Sort? Between Hoares and Lomato which partition scheme is faster? These are the questions that will be addressed.

For the testing of which variety of QuickSort is faster I choose the Java programming language to answer this question. I found Java versions of QuickSort basic, QuickSort with median3, QuickSort with Lomato, Quicksort with Lomato as well as a QuickSort with Insertion that was written in C# that I rewrote in Java that were used to test which variety of QuickSort is faster and the references to these can be found in the code reference section at the end of the document.

The included Testing.java is the file that I wrote to test the Java implementations of QuickSort for runtime(how long it takes to do the sort) when imputing various array(a collection of elements) sizes as well as I used it to test for the most time efficient Quicksort region size when it is using Insertion Sort. In an effort to gauge the speeds of the QuickSort implementations I inputted arrays into the QuickSort implementations using the Testing.java that were filled with random integers from 0 to 200 that were generated using Java’s Random function and averaged them together. 40 different array sizes were used and the arrays ranged from a size of 500 up to 1,000,850 in increments of 25,650. The QuickSort with Insertion Sort was tested on regions from 11-20. The time outputs of the QuickSort implementations using various array sizes were then copied from the terminal and inputted into an Excel file where there average running speeds were calculated for the arrays and that Excel file is included on the following three pages of this report. Following the excel report is my interpretation of the data with respect to the questions of: Does the median of 3 make QuickSort run faster? Is there an optimal region size to first run QuickSort on followed by Insertion Sort? Between Hoares and Lomato which partition scheme is faster?

When looking at the averages of the time taken in milliseconds of each QuickSort method from the excel spreadsheets one can see that on average by running the median of 3 it is over 10 times faster than running the basic version of quicksort. The spreadsheet’s averages also reveal that on average the fastest QuickSort with Insertion sort has a region size of 13. The spreadsheets also revealed that on average Hoares sorted in about a third of the time that it took for Lomato to do the same sorting. Overall the sorting speed of various QuickSort methods varied greatly in this study, with the fastest one being the median of 3 and the slowest one being the QuickSort with Insertion Sort using a region of 17.

General References:

“Quicksort.” Wikipedia, Wikimedia Foundation, 16 Sept. 2019, <https://en.wikipedia.org/wiki/Quicksort>.

Code References:

“Hoare's vs Lomuto Partition Scheme in QuickSort.” *GeeksforGeeks*, 21 Sept. 2019, <https://www.geeksforgeeks.org/hoares-vs-lomuto-partition-scheme-quicksort/>.

“Quicksort / Insertion Sort Combo.” *Coding Notes*, 3 May 2009, <https://megocode3.wordpress.com/2008/01/28/8/>.

*Quicksort: Simple Version of Quick Sort : Sort " Collections " Java Tutorial*, <http://www.java2s.com/Tutorial/Java/0140__Collections/Quicksortsimpleversionofquicksort.htm>.

*Quick Sort with Median-of-Three Partitioning : Sort " Collections " Java Tutorial*, [www.java2s.com/Tutorial/Java/0140\_\_Collections/Quicksortwithmedianofthreepartitioning.htm](http://www.java2s.com/Tutorial/Java/0140__Collections/Quicksortwithmedianofthreepartitioning.htm).