**Assignment 6**

Hits as a Time Predictor

**Name:** Hariharan Sundaram

**NUID:** 002915360

**Task**

Determine-for sorting algorithms-what is the best predictor of total execution time: comparisons, swaps/copies, hits (array accesses), or something else.

Run the benchmarks for **merge sort**, **(dual-pivot) quick sort**,and **heap sort**. Sort randomly generated arrays of between 10,000 and 256,000 elements (doubling the size each time).

**Execution**

The task was performed using elements starting from 8000 to 256000, doubling the size of the array each time. A random number array of integers was generated to perform the task using the Helper class. *SortStatsBenchmark* class was created to run the experiment 32 times for each array size.

Basic Merge Sort, Dual Pivot Quick Sort and Heap Sort were used in this task for comparison. In each run, swaps, compares, copies and hits were calculated using the InstrumentedHelperClass along with the average run time for each. Finally, these values were plotted for comparison. The experiment was run twice, once with instrumentation and once without. Following are the metrics and plots computed –

**Observations**

Text

Description automatically generated

Chart, line chart

Description automatically generatedChart, line chart

Description automatically generatedChart, line chart

Description automatically generated

From the above graphs and metrics, we can observe that the ***run time is closest correlated with the number of array accesses or hits***.

1. In case of heap sort, the number of copies is zero and hits are accessing the array created to represent the heap in memory.
2. In case of merge sort, the array accesses increase with the merge process and insertion sort on cut off. As the array size increases, the hits also increase and hence the time.
3. In case of dual pivot quick sort, the array accesses happen during the partitioning run comparing with the pivots. This also increases number of hits with the input array size and also increases the run time.

Copies and Swaps are cheaper compared to Data-Comparisons because they do not need array accesses and can be performed with just the memory address of a data point.

Data hits is the most expensive operation as the data point needs to be accessed from the heap. Most times with large data, there could be a storage access which is even slower compared to memory access. This is a big bottle neck deciding the run times of applications.

**Thus, we can conclude that number of Hits or Memory access is the most important factor deciding the run time of algorithms.**