DOKUZ EYLUL UNIVERSITY
ENGINEERING FACULTY
DEPARTMENT OF COMPUTER ENGINEERING

CME2204 ALGORITHM ANALYSIS ASSIGNMENT REPORT

Comparison of Heapsort, dualPivotQuicksort and ShellSort

by
Rıdvan Özdemir
2017510086

Lecturers
Dr. Zerrin Işık
Ali Cüvitoğlu
Ezgi Demir

IZMIR
19.04.2020

CHAPTER ONE

PROGRESS DESCRIPTION

The aim of the project is to develop the implementation of heapsort, dualPivoQuicksort and shellsort algorithms and testing these algorithm on different size and types of data. According to these tests results, sort algoritms are compared each other. Finally, algorithms time complexity in Big O notation and real running times are compared and appropriate algorithm is implemented in a scenerio.

CHAPTER TWO

TASK SUMMARY

2.1. Completed Tasks

maxHeap method were implemented. This method converts the data to maximum heap.

Heapsort were implemented. This method sorts the data by using heapsort algorithm.

Partition method were implemented. This method takes the left pivot and right pivot of to use in the quicksort method.

DualPivotQuicksort method were implemented. This method sorts the data by using dualPivotQuicksort algorithm

shellsort method were implemented. This method sorts the data by using the shellsort algoritm.

Time class were created and its all necessary methods were implemented to calculate running time of algorithms.

All necessary tests were applied for each sort algorithm.

2.2. Incomplete Tasks

There is no incompleted task.

CHAPTER THREE

EXPLANATION OF ALGORITHMS

3.1. Algorith and Solution Strategies

First of all, I implemeted the heapsort algorithm. In this step, I convert the data set to maximum heap by comparing the data in the array and the biggest data were put to last index of array. Then, I repeated this step recursively for this array except its sorted elements. Secondly, I implemented the dualPivotQuicksort algorithm. In this step, I divided the data set according to left pivot(first element of the data set for the beginning) and the right pivot (last element of the data set for the beginnig). After these step, I used the dualPivotQuicksort method recursively to sort the data set by using these left pivot and right pivot. After that, I implemented shellsort algorithm. In this step, I determined the gap as [N/4, N/2, ……, 1]. Then, I applied the shellsort algorithm by using the gap. Then, I created a Timer class to calculate the running time of the sort algorithms. Finally, I tested the algorithms for specific size and type of date set.

CHAPTER FOUR

PROBLEMS ENCOUNTERED

Firstly, I had to store and return the left pivot and right pivot values to use in dualPivotQuicksort method. But this is impossible in a single method. A method only returns one value. To store and return these values, I stored them in an array and method returned this array.

Then, I encountered stackoverflow exception in dualPivotQuicksort for 100.000 sized increasing and decreasing integers data set. Firstly, I tried use the garbage collection by using the try-catch block to solve this problem. But it did not work, because when the program entered the catch block, sorting process end. After the a bit researching on the internet, I solved this problem by giving some arguments to JVM.

CHAPTER FIVE

CONCLUSION

	EQUAL INTEGERS			RANDOM INTEGERS			INCREASING INTEGERS			DECREASING INTEGERS		
	1.000	10.000	100.000	1.000	10.000	100.000	1.000	10.000	100.000	1.000	10.000	100.000
heapSort	0,178 ms	4,741 ms	7,023 ms	0,188 ms	2,405 ms	17,336 ms	0,147 ms	1,017 ms	10,666 ms	0,104 ms	1,023 ms	7,012 ms
dualPivotQuicksort	1,032 ms	3,916 ms	10,197 ms	0,108 ms	2,434 ms	16,734 ms	1,946 ms	20,910 ms	2326,4 ms	2,552 ms	20,952 ms	2356,7 ms
shellSort	0,836 ms	3,134 ms	9,215 ms	0,627 ms	7,453 ms	16,353 ms	0,461 ms	0,155 ms	3,142 ms	0,345 ms	0,428 ms	2,664 ms

Table 1. *Performance matrix*

Time complexity of buildHeap is O(n) and overall time complexity of heapsortis O(nlogn) for all cases.That's why totally, it takes much more less time than other sorting algorithms. Also even the dualPivotQuicksort has O(nlogn) time complexity on average but in the worst case its time complexity is O(n^2). That's why, in the worst cases as increasing and decreasing integer it takes much more time than other algorithms. Finally, shellsort has O(nlogn) in best cases and O(n^2) time complexity in worst cases. That's why it takes much more less time than dualPivot in increasing order because, increasing order is the best case for shell sort. Also random integers is the worst case for shellsort. That's why it takes much more time in this situation. These results is also seen in the performace matrix. Values of the table may change a little according to computer's and memory's processes

The appropriate sorting algorithm for following scenerio is heap sort. In the following scenerio, we must sort the one million student's central exam grades. These grades are random numbers, each student has different grade. According to the performance matrix and the time complexities, heap sort algorithm will be best choice for this scenerio.

REFERENCES

https://www.geeksforgeeks.org/shellsort/

http://bilgisayarkavramlari.sadievrenseker.com/2008/12/20/kabuk-siralama-shell-sort/

http://bilgisayarkavramlari.sadievrenseker.com/2008/08/09/yiginlama-siralamasi-heap-sort/

https://stackoverflow.com/questions/12767588/time-complexity-for-shell-sort

Data Structures and Algorithm Analysis in Java, Clifford A. Schaffer