Comparison of Merge sort, Heapsort and Quicksort

Algorithm Analysis

by

Gözde Işıldak

25.03.2018

İZMİR

PURPOSE

The purpose of this project is to implement implement and test Merge Sort ,Quick Sort , and Heap Sort algorithms and determine the appropriate sorting algorithms for the following scenarios.

INTRODUCTION

The sorting algorithms used in this project:

1)MergeSort

- TwoParts : recursively split the array into 2 parts
- ThreeParts : recursively split the array into 3 parts

2)HeapSort

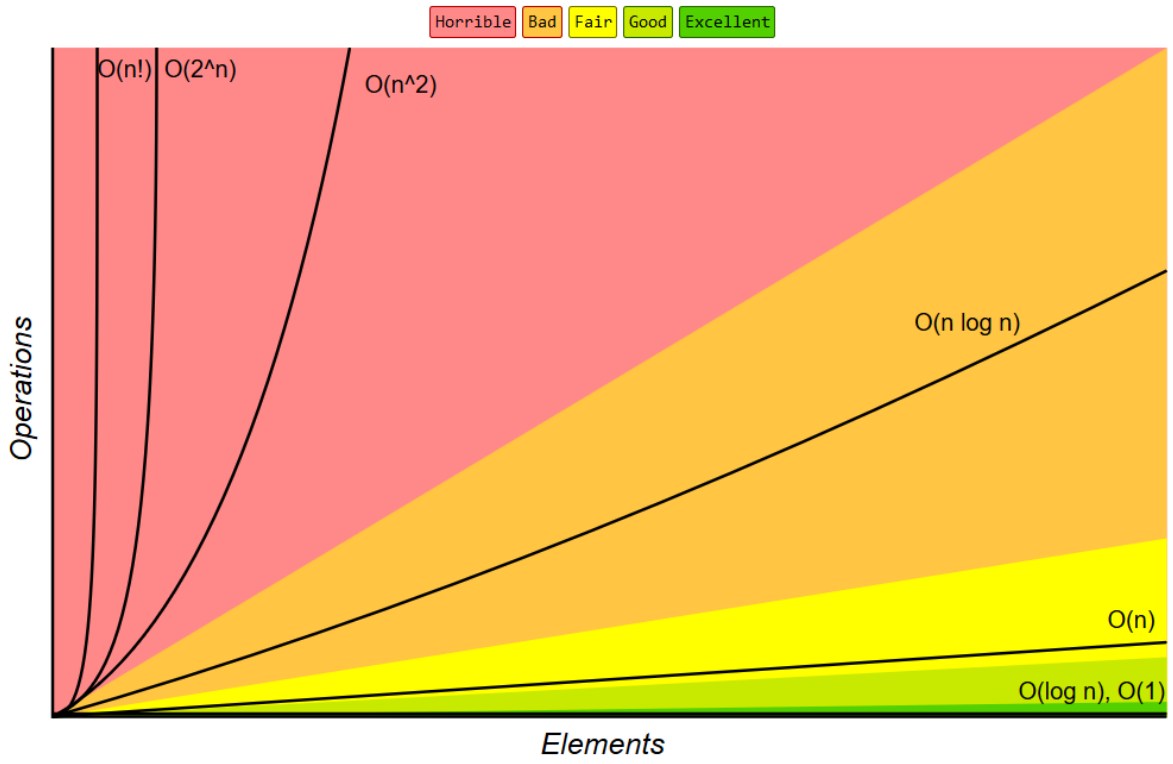
3)QuickSort

- FirstElement : The pivot is always the first element of the array to be sorted.
- RandomElement : The pivot is chosen at random from any element in the array to be sorted.
- MidOfFirstMidLastElement : The pivot is chosen to be the element whose value is in the middle among the {first, middle, and Last} elements in the array to be sorted.

Of course, a certain amount of time passes for these algorithms to work. In the following table these 3 algorithms are given time complexity.

Algoritms	Time Complexity		
	Best	Average	Worst
QuickSort	Ω(n log(n))	Ω(n log(n))	Ω(n*2))
MergeSort	Ω(n log(n))	Ω(n log(n))	Ω(n log(n))
HeapSort	Ω(n log(n))	Ω(n log(n))	Ω(n log(n))

Big-O Complexity Chart



Comparison of Merge sort, Heapsort and Quicksort

In this assignment 3 sorting algorithm was tested with 1000,10000,10000 elementary arrays and the times were recorded in the following table.

	EQUAL INTEGERS			RANDOM INTEGERS			INCREASING INTEGERS			DECREASING INTEGERS		
	1000	10000	100000	1000	10000	100000	1000	10000	100000	1000	10000	100000
MS two	957235	3362374	26794303	783408	2926226	21910150	750618	2335213	14444659	772347	2474669	17802293
MS three	610371	3628251	17886836	598914	2834571	18198145	453926	2663114	8735220	481976	2825880	11482878
HS	656593	3086621	21524964	538470	2856300	16995575	617087	2627558	15292463	710717	2955065	18689207
QS first	571655	3440992	21605556	552692	2843657	13419868	4495017	24298695	1683813551	5434081	53231075	7321656480
QS rnd	924840	6293736	19100072	775112	6038526	18956665	794865	6701835	18579380	779063	6072106	16683477
QS fml	528988	4678327	19723480	526617	2832991	12658978	425482	1257484	3436251	583506	3123362	14930191

Data is already sorted at increasing integer array and decreasing integer array,worst case behavior occur at QuickSort Algorithm first element and the worst running time is O(n^2).QuickSort performance is dependent on pivot selection algorithm.But if pivot is choosen randomly or median of three(first,last and middle),best case or average case occur.Best and average running time are O(n * log(n)).When randomly selected,its possible to construct data that results in worst case behavior,however it's rare for it to come up in normal usage.

Regular merge sort is n * log_2(n) which is equivalent to n * (log(n) / log(2)). The log(2) is constant, so merge sort is simply n * log(n)

Tri-merge sort is n * log_3(n) which, using the same logic for regular merge sort, is simply n * log(n)

Given that both reduce to O(n * log(n)), it's not really possible to say which is better.

TriMergeSort is faster than that of MergeSort according to the table.

The running time of HeapSort is O(n * log(n)) and this is fast enough according to the table.

SCENARIO 1

You have a Turkish-English dictionary with a single word for each word in alphabetical order and you want to translate it into an English-Turkish dictionary. For example; if your Tur-Eng dictionary contains [ayı : bear, bardak : glass, elma : apple, kitap : book] then your Eng-Tur dictionary should be [apple : elma, bear : ayı, book : kitap, glass : bardak].

If we think that there are thousands of words in your dictionary, which sorting algorithm do you use to do this translation faster?

- We can use MergeSort ThreeWay, QuickSort MidOfFirstMidLastElement or FirstElement because Turkish words are arranged randomly while we sort words. This situation is seen in table.

SCENARIO 2

When you inquire Sub-Upper Pedigree, an ordered list of people according to their birth date comes out in the system of e-Devlet. However, you want to rank the people from the youngest to the oldest one. If you are asked to do this operation using a sorting algorithm, which algorithm do you use?

- We can use MergeSort ThreeWay, QuickSort MidOfFirstMidLastElement. This situation is seen in table.