DOKUZ EYLUL UNIVERSITY

ENGINEERING FACULTY

DEPARTMENT OF COMPUTER ENGINEERING

CME2204 ALGORITHM ANALYSIS REPORT

ASSIGNMENT – I

COMPARISON OF HEAPSORT, SHELLSORT AND INTROSORT

by

Ali Şiyar Arslan

2019510017

IZMIR

06.04.2022

	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	1	2	4	1	2	12	0	1	9	1	2	9
shellSort	0	2	8	1	3	14	1	2	8	1	2	8
introsort	1	3	13	0	2	12	0	1	8	1	2	13

Comparison

İf you have equal , a lot integers you should use heap sort

İf you have random , a few integers you should use introsort

İf you have decreasing, a few or average integers you can use any sorting algorithms

İf you have random integers you can use heap or intro sort algorithms

Scenario:

Shell sort because when integers are random and big size result of my code shell sort is the most fast algrotihm.

Introduction

Sorting is a fundamental operation in computer science. Sorting generally refers to the operation of ordering data in a given sequence such as increasing sequence or decreasing sequence. There are many fundamental and advance sorting algorithms. All sorting algorithm are problem oriented means they work well on some special problem and do not work well for any kind of a problems. A sorting algorithm is used to arrange elements of an array/list in a specific order. Sorting is the process of arranging the elements of an array so that they can be placed either in ascending or descending order. For example, consider an array A = {A1, A2, A3, A4, ?? An }, the array is called to be in ascending order if element of A are arranged like A1 > A2 > A3 > A4 > A5 > ? > An .

Heap sort

One of the disadvantages of heap sort is that it works slower than other sorting methods with same computational complexity. Moreover, it is not efficient for parallelization. However, heap sort is considered often for large data sets for the reason that it does not work recursively all the time. Sorting data stored in linked list data structure, heap sort is not recommended due to the fact that it's difficult to convert the linked list to heap structure. The heap sort belongs to a selection sort. Firstly, to structure the record sequence to be sorted into a heap, then select the 1st maximal records in the heap, remove it from the heap, and readjust the rest records into a heap so that the 2nd maximal record is found and so forth till there is merely one record in the heap. This process is known as a heap sort. Time complexity: the main time spent in the heap sort is in repeated screening during the initial heap construction and new heap

adjustment. For the heap sort, the time complexity at the worse case is O (nlog2n). This is the biggest advantage of the heap sort.

Shell sort.

Shell sort belongs to an insertion sort. The record sequence to be sorted is divided into several groups. Within each group, the records are separately processed with straight insertion sort so that the overall record sequence are partial sequential; the procedure is repeated till all the records are in one group. Finally, all records are processed with a straight insertion sort. Time complexity: the time performance of Shell sort is between O (n2) and O (nlog2n). When n falls within a certain range, the necessary comparison times and record moving times are approximately O (n1.3) in Shell sort.

Introsort

Intro sort is an efficient in-place sorting algorithm, which usually beats all other sorting algorithms in terms of performance. Due to its high performance, it is used in several standard library sort functions, including some C++ sort implementations. Introsort is a comparison sort, meaning that it can sort items of any type for which a less-than relation is defined. It is usually not a stable sort. It is a hybrid sorting algorithm that uses Insertion sort, Quick sort and Heap sort to maximize efficiency. Introsort or introspective sort is a hybrid sorting algorithm that provides both fast average performance and (asymptotically) optimal worst-case performance. It begins with quicksort, it switches to heapsort when the recursion depth exceeds a level based on (the logarithm of) the number of elements being sorted and it switches to insertion sort when the number of elements is below some threshold. This combines the good parts of the three algorithms, with practical performance comparable to quicksort on typical data sets and worst-case O(n log n) runtime due to the

heap sort. Since the three algorithms it uses are comparison sorts, it is also a comparison sort.

Critical ideas to think

Shell sort

The best case for shell sort, is when the array is already sorted. Then the number of comparisons for each of the increment-based insertion sorts is the length of the array. The point about shell sort is that the initial sorts, acting on elements at larger increments apart involve fewer elements, even considering a worst-case scenario. At the larger increments, "far-away" elements are swapped so that the number of inversions is dramatically reduced. At the later sorts using smaller increments, the behavior then comes closer to optimal.

İntro sort

Introsort uses a suitable sorting algorithm depending upon the data set. Since insertion sort is good with more minor data, it sorts that kind of data. Quicksort is good when the pivot element is an intermediate value obtained by finding the median. And heapsort is used when the iterations get higher. Still, introsort has the main disadvantage that it cannot put data into the array. Uses quick sort when the pivot is median and heapsort for sorting large dataset.

Heap sort

independent of distribution of data. heapsort is used when the iterations get higher. While other sorting algorithms may grow exponentially slower as the

number of items to sort increase, the time required to perform Heap sort increases logarithmically. This suggests that Heap sort is particularly suitable for sorting a huge list of items. Furthermore, the performance of Heap sort is optimal. This implies that no other sorting algorithms can perform better in comparison.

References

Lokesh Karthikeyan(25 Nov, 2021) *IntroSort or Introspective sort* Geeksforgeeks https://www.geeksforgeeks.org/introsort-or-introspective-sort/

(17 Aug 2021) *ShellSort* Geeksforgeeks https://www.geeksforgeeks.org/shellsort/

(17 jan 2022) *HeapSort* Geeksforgeeks https://www.geeksforgeeks.org/heap-sort/