Assignment 2
Algorithms and Data Structures 1 (1DL210)
2021

Louise Andersson, Sara Lindberg, Ellen Siggstedt, Mathilda Stigenberg

October 5, 2021

Comparison

What are the differences between the three algorithms?

Answer:
Insertion Sort is an algorithm which divides an array into two sub-arrays. During the process one of the sub-arrays is sorted and the other is unsorted. Initially the whole array is unsorted, but for each iteration of the algorithm an additional value is added to the sorted sub-array. The execution is done when the whole array is sorted.

Quicksort is also an algorithm which divides an array into two subarrays. The difference from Insertion Sort is that Quicksort uses the "Divide-and-Conquer" method and sorts the array recursively.

Heapsort also uses recursion when it is sorting an array, at least partially in the algorithm. The difference between Heapsort and the other two methods is that it uses a different data structure, namely heap. When Heapsort is sorting the array, it handles the array as a tree structure, where it compares the children nodes to the parent node and switches them if one child has a higher value than the parent node.

All three of the algorithms are in place algorithms.

For each algorithm, mention a situation where it has an advantage over the other two.

Answer:

Insertion sort

An advantage with Insertion sort is that it performs well in the case when the array is already sorted, with a complexity of $\Theta(n)$. In that case this algorithm is faster than for its worst- and average case where the complexity is $\Theta(n^2)$. Unlike Quicksort and Heapsort which performs with the complexity of $\Theta(nlog(n))$ respectively $O(nlog(n))$ in both best- and average case. Heapsort even performs with the same complexity for worst case as for best case. This means that Insertion sort is the only of these three algorithms that actually has an advantage when the array already is sorted.

Quicksort

Quicksort is heavy for small sized arrays but has an advantage when it comes to very large sized arrays. Furthermore, Quicksort is often the best choice to use for sorting since it is very efficient in the average case. In the average case the expected running time is $\Theta(nlog(n))$

Heapsort

In the case of Heapsort an advantage with the algorithm is that even when the array is inverted sorted, in other words the worst case, it has the same complexity of $O(nlog(n))$ as in other cases. This means that the array can be the best, worst or average case and the complexity is always the same. That makes the algorithm quite predictable. Compared to Quicksort and Insertion Sort, which have a rather poor complexity in the worst case of $\Theta(n^2)$.