Algorithm PA1 Report

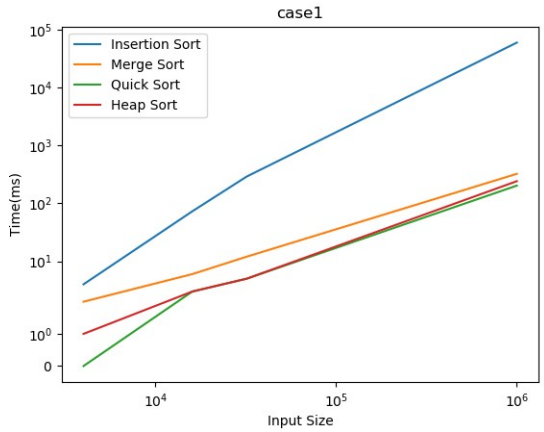
B07901069 電機二 劉奇聖

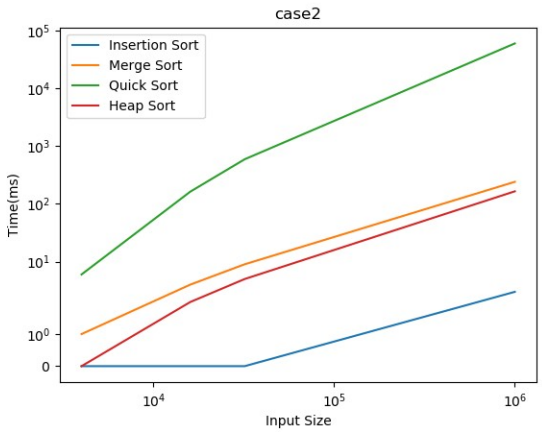
1. Running time table

Input size	IS		MS		QS		HS	
	CPU time(ms)	Memory(KB)	CPU time(ms)	Memory(KB)	CPU time(ms)	Memory(KB)	CPU time(ms)	Memory(KB)
4000.case2	0	12500	1	12500	5.999	12500	0	12500
4000.case3	9.999	12500	1	12500	5.999	12520	1	12500
4000.case1	4	12500	2	12500	0	12500	1	12500
16000.case2	0	12648	4	12648	162.975	12648	2	12648
16000.case3	144.978	12648	4	12648	164.975	12948	2	12648
16000.case1	72.989	12648	5.999	12648	3	12648	3	12648
32000.case2	0	12648	8.999	12648	592.91	12648	5	12648
32000.case3	586.911	12648	7.999	12648	527.919	13308	3.999	12648
32000.case1	290.956	12648	11.998	12648	4.999	12648	4.999	12648
1000000.case2	2.999	18668	240.963	20524	too long	N/A	146.978	18668
1000000.case3	too long	N/A	246.962	20524	too long	N/A	107.984	18668
1000000.case1	too long	N/A	323.952	20524	202.97	18668	241.963	18668

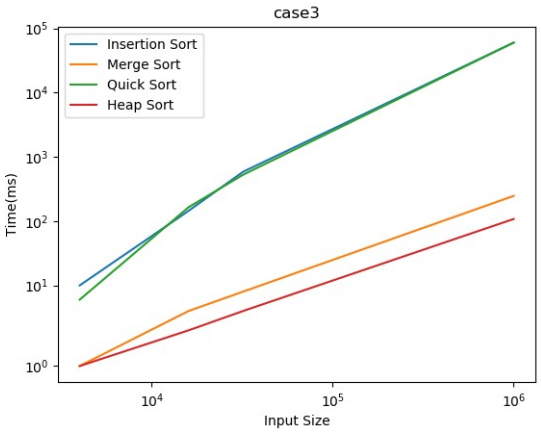
2. Running time curves

(Assume the CPU time for “too long” is 60000 ms (1 min). For clearity, I take symmetrical log scale for input size and running time.)





In case1(random order), insertion sort is the slowest, it is a direct result that insertion sort runs in $O(n^2)$ in average. And we can find that quick sort is the fastest among three O(n lg n) algorithms.



In case2(sorted order), insertion sort is the fastest because it is O(n) in this case. Quick sort is the slowest because it is the worst case, which is $O(n^2)$, for quick sort.

In case3(reversed sorted order), insertion sort and quick sort are all slow because they are $O(n^2)$.