z5242692

Chenqu Zhao

COMP9101 (T2-2020)

Homework 1 - Q1

(a) According to the combination rule, the n distinct integers in array A can form $\binom{n}{2} = \frac{n(n-1)}{2}$ unique pairs of integers $(\mathrm{A[k]}, \mathrm{A[m]})$, $k < m$. Calculate the sums of $\mathrm{A[k]}^2 + \mathrm{A[m]}^2$ for all $1 \leq k < m \leq n$ and put them in an array B of size $n(n\text{-}1)/2$. The multiplication costs $n(n\text{-}1)/2 = n^2/2 - n/2 = O(n^2)$.

Apply merge sort to sort array B, which costs $n^2 \log n^2 = n^2 \cdot 2 \log n = O(n^2 \log n)$.

Then, iterate the sorted array to check whether there are two adjacent equal values. The iteration costs $n(n\text{-}1)/2 = n^2/2 - n/2 = O(n^2)$.

The total cost is $O(n^2) + O(n^2 \log n) + O(n^2) = O(n^2 \log n)$ which runs in time $n^2 \log n$ even in the worst case.

(b) Recall the same combination and multiplication operation from question (a), however, this time put the computed sums into a hash table instead of an array. This allows us to check if the sum already exists immediately when inserting it: each insertion and lookup takes $O(1)$ expected time.

For example, for sum x, we check if x is already in the hash table in $O(1)$. If so, the problem solved. Otherwise, insert it into the hash table also in $O(1)$.

The expected time cost of this algorithm is $O(n^2) \cdot O(1) = O(n^2)$.