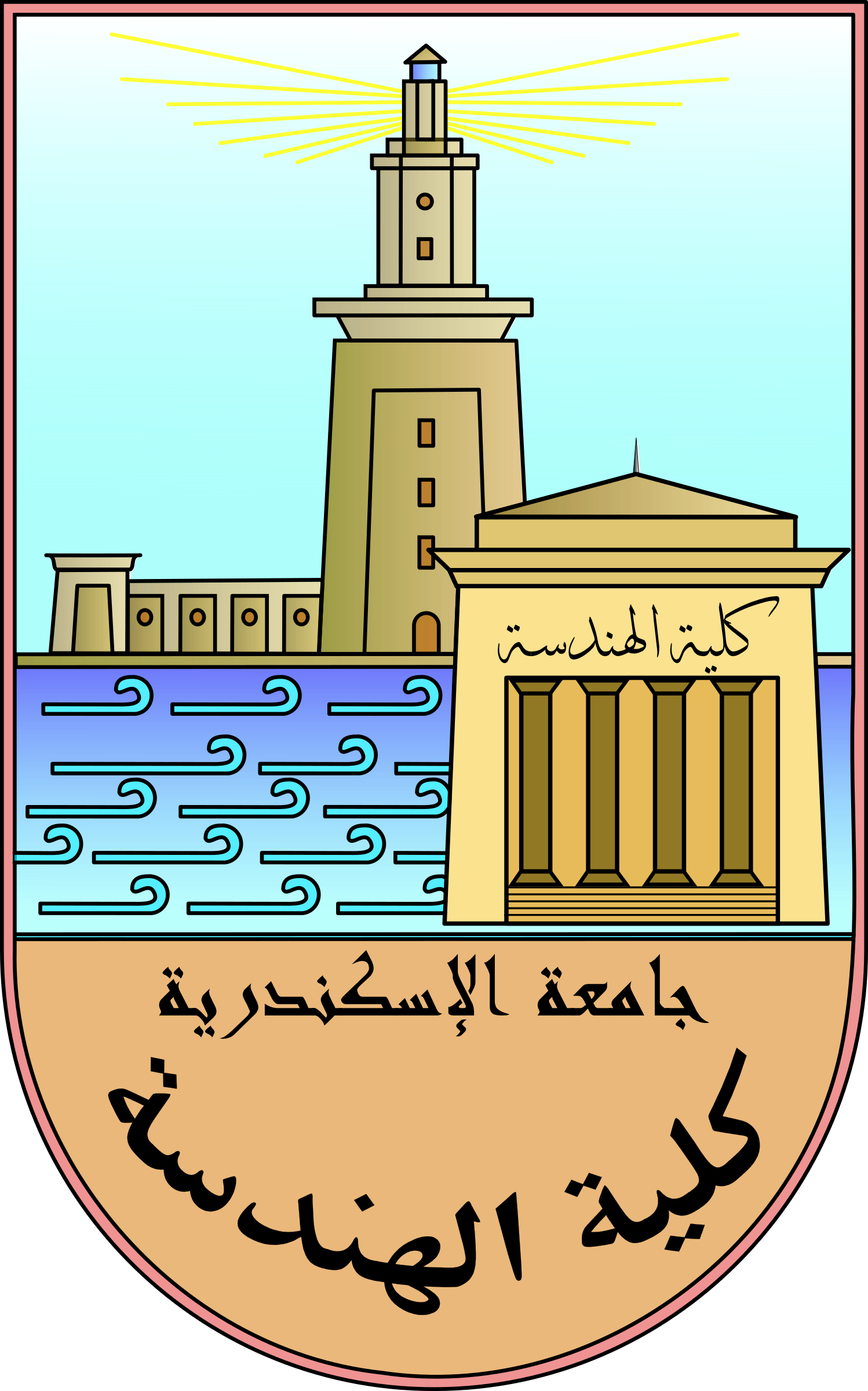
DSA: 3



Karim Tarek Ibrahim

Abdullah Taman

Yahya Azzam

Yousef Elmedany

1. Time Analysis

First of all, we’ll discuss the difference in insertion in both of the tables. In the O(N) insertion was done slower than O(N2), and that is because a lot of collisions (nearly 1/3 of the time) in O(N), but in O(N2) collisions occur very rarely (1/100000).

Second thing was deletion, O(N2) was also faster in deletion because it requires searching in one contiguous table, while in O(N) requires searching in N tables.

Third thing is searching, which is better in O(N2) also, because one hashing function is required to be evaluated, while two functions are required in the O(N), and it all happens in O(1).

Batch insertion and deletion inherit the same characteristics of insertion and deletion of both methods, but it inherits it \*n times.

The only thing that O(N) is superior to O(N2) is the space used, by using nearly 1/N than the O(N2), which could be used for larger sizes without any problem, where the size in O(N) is a maximum of 3.5\*M, where M = 2b, the smallest power of 2 larger than N, while in O(N2) the size is M, where M = 2b, the smallest power of 2 larger than N2­.

1. Comparison

You can find the whole comparison [here](file:///C:\Users\yahya\OneDrive\Desktop\coding\DSA_Perfect_Hashing_O_n__O_n2\Analysis.xlsx).

The O(N2) is much faster than the trees, while the O(N) is slower due to using small sizes, but on increasing the size to more than 10^6 elements, the trees become much slower and inefficient.

Alhamdullilah.

That’s it, thank you.