534575-2-7KE AID: 258164 | 30/07/2020

The difference between unsorted optimized array and sorted array structure are as follow:

**Unsorted Optimized Array:**

All elements of the array are initialized to null for this, any other array-based structure.

Insert : In Insert uses the first null array element to store the new nodes location. No loop, 3 memory access ,O(1).

Fetch : Fetch uses a sequential search. In worst case, Search average time: n/2 ,2 memory access , O(n).

Delete: In Delete ,uses a sequential search moves up all nodes below the deleted node. Search average time: n/2 ,2 memory access ,O(n)

Update: n +3 memory access , O(n)

**Sorted Array:**

All elements of the array are initialized to null. Nodes are stored in sorted order based on key.

Insert : Places the inserted node in its sorted order position. After moving larger nodes references down in array. Search average time: n/2 , 2 memory access ,O(n).

Fetch : Fetch uses a binary search. In worst case, Search average time: log2 n , 2 memory access , O(log2n)

Delete: Delete uses a binary search. moves up all node references below the deleted node. Search average time: n/2 , 2 memory access ,O(n)

Update: 2n memory access ,O(n).

Conclusion:

From the above difference conclusion is that the average memory access of sorted array > unsorted optimized array . Due to insertion and fetch operation the search average time makes unsorted optimizes array faster than sorted array structure. It is faster to process a sorted array than an unsorted array because of branch prediction. In CA, a branch prediction determines whether a conditional branch (jump) in the instruction flow of a program is likely to be taken or not.