Work Progress

kNN Search with Parallel Incremental Query Answering

Jaouhara Chanchaf

Tuesday Feb 8th, 2023

1. Summary

Done:

AI 1	- Adapt Min-Max heap C++ code to store and manipulate kNNs. - Test Kashif performance when using Min-Max Heap instead of the sorted array.

In progress:

AI 2	Search for batch insert to Min-Max Heap.

2. Kashif: Average query time, storing kNNs in a sorted array vs OS-Tree

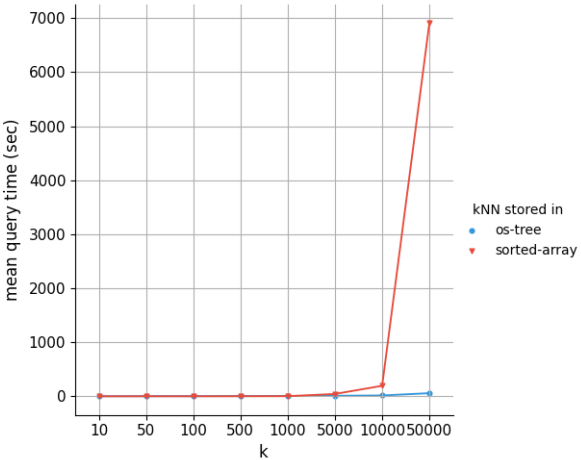

Figure 1: Kashif mean query time
(10 queries, query size = 100, dataset = 100k tables, 490k cols, 5M vectors)

2. Kashif: Average query time, storing kNNs in a sorted array vs OS-Tree

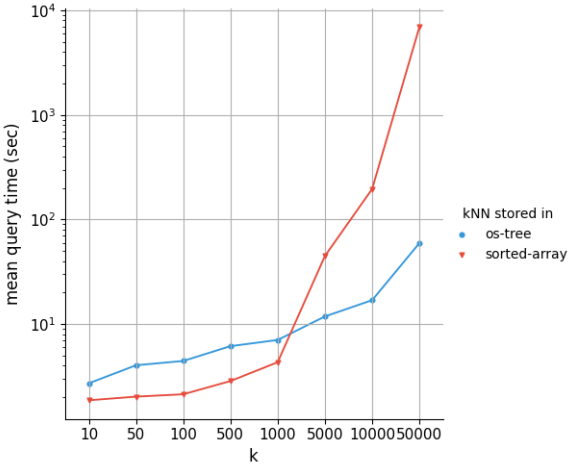

Figure 2: Kashif mean query time (log scale)
(10 queries, query size = 100, dataset = 100k tables, 490k cols, 5M vectors)

3. Discussion

- Min-Max Heap does not support Select(i) Operation (must pop the ith element to get the (i+1)th element.
- Used Order Statistics Tree (OSTree) to store kNN results. OSTree supports Select(i) in logarithmic time.

Worst Case Time Complexity:

	Sorted Array	Order Statistics Tree
Insert(d)	$O(n^2)$	$O(log(n))$
Select(i)	$O(1)$	$O(log(n))$