ISSN 2278-3091
Volume 10, No.5, September - October 2021
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse041052021.pdf
https://doi.org/10.30534/ijatcse/2021/041052021

Testing Performance (Time Analysis) of Nearest Neighbour (NN) Search Algorithms on K-d Trees

Mihir Goyenka[1], Ayush Kedia[2]
[1]Department of Computer Science and Engineering, Shri Ramdeobaba College of Engineering and Management, Nagpur, India,mihirgoyenka@gmail.com
[2]Department of Computer Science and Engineering, Shri Ramdeobaba College of Engineering and Management, Nagpur, India,ayushk05@gmail.com

Received Date : August 06, 2021 Accepted Date : September 15, 2021 Published Date : October 06, 2021

ABSTRACT

K-d tree (k-dimensional tree) is a space partitioning data structure for organizing points in a k-dimensional space. K-d tree, or Multidimensional Binary Search Tree is a useful data structure for several applications such as searches involving a multidimensional search key (e.g., Range Search and Nearest Neighbour Search). K-d trees are a special case of binary space partitioning trees.KNN Search is a searching algorithm with complexity O(N log N) {N= no. of data points}. This search algorithm is relatively better than brute force search {Complexity= O(n*k); where k=No. of neighbours searched, N=No. of Data Points in Kd tree} for dimensions $N>>2^D$ {N=No. of Points, D=Dimensionality of Tree}.Furthermore, Parallel KNN Search is much more efficient and performs better than KNN Search, as it harnesses parallel processing capabilities of computers and thus, results in better search time.This paper tests the time performance of KNN Search and Parallel KNN Search and compares them by plotting it on a 3D graph. A more comprehensive comparison is done by use of 2D graphs for each dimension(from 2 to 20).

Key words: KNN search, parallel knn search, dimensions, datapoints.

1. INTRODUCTION

KNN search is a type of NN search algorithm wherein 'K' is the nearest neighbours found. In this paper, K=15 is used for significant value of search time. Furthermore, Parallel KNN search is a faster NN search algorithm than KNN search as it utilizes parallel processing, which is faster than conventional methods of computation. Thus, for systems with more and powerful processors, Parallel KNN Search algorithm is relatively easily executable for extremely large datasets.

2.METHODOLOGY

2.1. Building Kd Tree

Building a Kd tree has following possible complexities:

$O(N \log^2 N)$

O(N logN)

O(K(N logN))

After building the tree, both algorithms are separately tested on it with the following specifications:

1. K=15 (No. of neighbours searched in all test cases).
2. Dimensions: {2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20}
3. No. of Data Points ϵ {5000,10000,15000……150000}

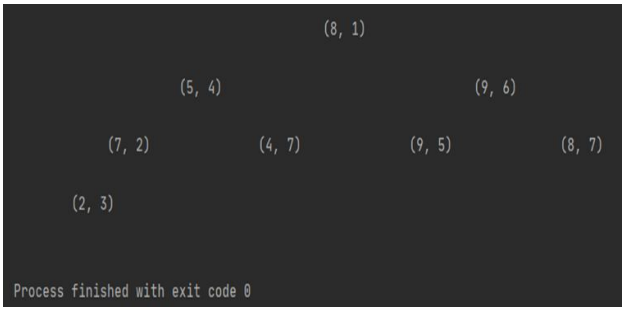

Figure 1. Sample Kd Tree generated by Python module

Sample KD Tree node generated via ***NumPy*** and written to text file (dim=12):

[(None, None, array([[5.80824347e-05, 4.14702495e-03, 2.98221226e-03, 1.62450649e-04, 2.39736370e-04, 6.11501799e-05, 2.17134063e-03, 9.02927112e-04, 7.48986886e-04, 6.00517300e-03, 8.81551468e-05, 5.77593033e-04]

As the data to be processed increases along with dimensions, ***CurseofDimensionality*** starts slowing the algorithm down, causing excessively lengthy search time, which can be undesirable.

Degradation of performance also occurs when the average pair-wise distance between K nearest neighbours is significantly less than distance between the node on tree and the point to be searched.

Established studies show that Quadratic search (Brute Force Search) performs equally well as KNN Search and Parallel KNN Search for smaller datasets and lower dimensions.

Table1: Search Analysis Data

Dimension (Ndim>=2)	No. of Data Points(N)	No. of neighbours searched	KNN search Time(ns)	Parallel KNN Search Time(ns)	Time to build tree(ns)
	10000		1.94480242000E+10	6.2005650000E+09	
	25000		1.13097048800E+11	5.5516725700E+10	
	40000		2.50635900400E+11	1.3641144380E+11	
	55000		4.86134208100E+11	2.1971811820E+11	
10	70000	15	6.75784028400E+11	3.4633833290E+11	317177100
	25000		1.12583594500E+11	6.0406935900E+10	
	40000		2.77880860000E+11	1.4595701310E+11	
	55000		7.16700884700E+11	4.4409485330E+11	
11	70000	15	7.61988252200E+11	4.2051630880E+11	35870200
	25000		1.46835042800E+11	6.7567322900E+10	
	40000		3.07576537100E+11	1.6349422080E+11	
	55000		5.36608687300E+11	2.8624714720E+11	
12	70000	15	8.71397389400E+11	4.6139401680E+11	336100900
	10000		2.30932720000E+10	1.2343049400E+10	
	25000		1.24209596500E+11	6.4844150300E+10	
	40000		4.58700668500E+11	2.4147242250E+11	
13	55000	15	7.90384821100E+11	4.4498562470E+11	294210300
	10000		2.31640858000E+10	8.0219469000E+09	
	25000		1.27391884500E+11	7.2069365800E+10	
	40000		3.33065692500E+11	1.8136142710E+11	
14	55000	15	6.78884679300E+11	3.2297809110E+11	181515200
	10000		2.444868420000E+10	1.1068116500E+10	
	25000		1.271764407000E+11	6.8335651500E+10	
15	40000	15	3.940109276000E+11	1.8426735540E+11	38864600

COMPONENTS USED (Python Libraries):

- Matplotlib for data visualization and graph plotting
- Multiprocessing, Ctypes and OS for Parallel processing
- NumPy for data organizing and manipulation for efficient handling.
- Time for timing the search algorithms

3.DISCUSSION AND RESULTS

Figure 2 provides a comparison for time taken for KNN search execution on KD tree for dimensions 2 to 20. The gradient of each line progressively increases with increase in dimensionality. Thus, search time for KNN search algorithm can be attributed to Dimension and No. of Nodes in K-D Tree

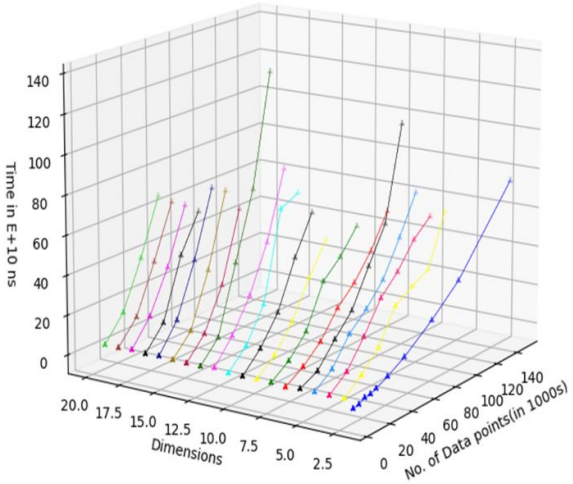

Figure 2. Plot of KNN Search Times

The Plot in Figure 4 visualizes Parallel KNN Search time for dimensions 2 to 20 and dataset range 20,000 to 140,000. Like KNN search, Parallel KNN search shows same characteristics for slopes of lines in increasing order of dimension and number of datapoints. However, it is considerably faster than KNN Search as plotted values along Z-Axis(Time) are significantly less.

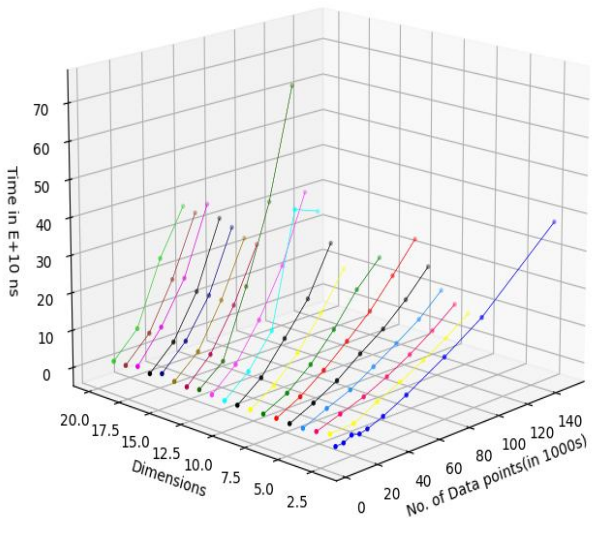

Figure 3. Plot for Parallel KNN Search Times

The graph in Figure 5 shows distinct differences in KNN Search and Parallel KNN Search Time values for same dimension and No. of Data Points. The corresponding slopes of lines within each dimension are also considerably varied for increasing no. of datapoints. The difference is much less pronounced for smaller datasets, which thus gives us a probable conclusion:

KNN Search and Parallel KNN Search algorithms have almost similar searching time for No. of DataPoints $\in$ {20,000 (±5,000) }

Thus, algorithms must be chosen accordingly.

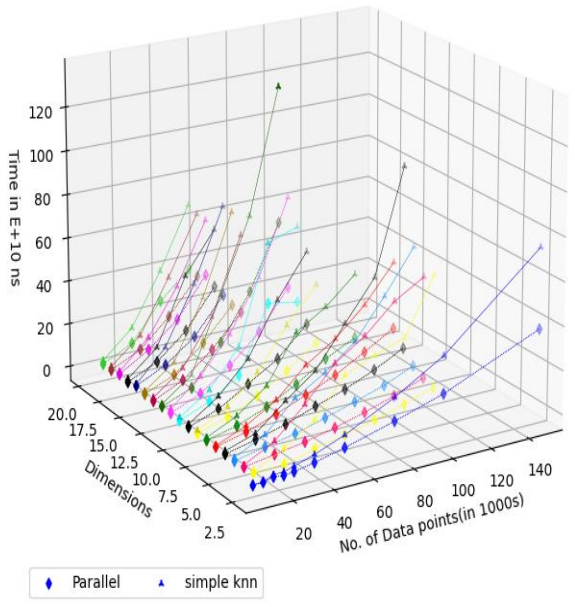

Figure 4. KNN vs Parallel KNN Search Time

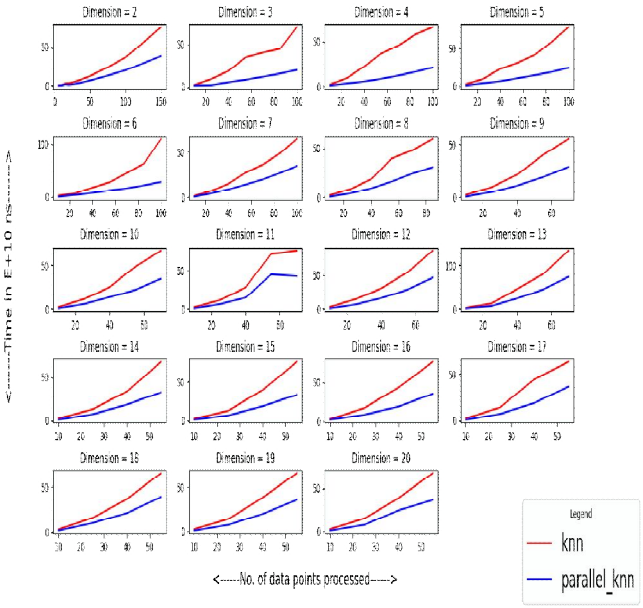

Figure 5.2D Plots for KNN vs Parallel Search Time Comparison

Dimension-wise plot for comparing knn search and parallel knn search times.

For each sub graph:

1 unit on X Axis = 1000 data points (number)

1 unit on Y Axis = Time in 10^{10} ns (time)

4.APPLICATIONS

This comparative analysis may be used for benchmarking in industries where KNN Search or Parallel KNN Search is to be used. Further, KNN algorithms are used in Machine Learning (Unsupervised) for classification purposes.

Robotics uses approximate NN search, which is a less strict algorithm than KNN Search, and is satisfied by the nearest node it finds.Thus, development of KNN search will greatly improve spatial recognition in Robotics.

KNN Search is also used in Information Retrieval algorithms, which have direct implications in Digital Library Systems, Search Engines, and Image retrieval systems.

5.CONCLUSION

The conclusions drawn from findings of this research paper are tabulated as follows:

1. KNN Search behaves in an identical manner to that of Parallel KNN Search when tested upon relatively small datasets(20000 points)

2. Since Parallel KNN Search relies on the compuatational power of the system on which it is executed (parallel processing & multi-threading), we conclude that optimization and improvement of performance of KNN Parallel search is Hardware dependant.

3. For extremely large datatsets, Parallel KNN Search time is significantly lower than that of KNN Search. Thus, when working with large datasets and dimensions, Parallel KNN Search must be used for reduced waiting time.

REFERENCES

1. Bulusu Rama *et al*, **Secure k-NN query on encrypted cloud data with multiple keys**, International Journal of Advanced Trends in Computer Science and Engineering, Vol. 8, No. 3, pp. 874-878, 2019.
2. J.L. Bentley,**Multidimensional Binary Search Trees used for AssociativeSearching**, Communications of the ACM Vol. 18, Number 9, Sept. 1975.
3.J.B. Rosenberg,**Geographical Data Structures compared: A Study of data structures supporting Region Queries**, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems Vol 4, Issue 1, Jan 1985.
4. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest**,Introduction to Algorithms**.MIT Press and McGraw-Hill, 1990.
5. Jacob E. Goodman, Joseph O'Rourke and Piotr Indyk, **Handbook of Discrete and Computational Geometry**, CRC Press LLC, Boca Raton, FL, 1997.