Indexing chemical fingerprints for efficient querying of molecular databases

A Project Report

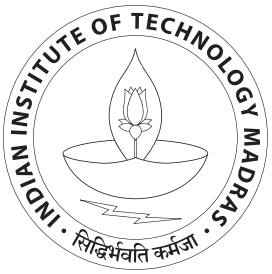
submitted by

ABHIK MONDAL

in partial fulfilment of the requirements
for the award of the degrees of

BACHELOR OF TECHNOLOGY
&
MASTER OF TECHNOLOGY

under the guidance of
Dr. Sayan Ranu



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY, MADRAS.

April 2015

THESIS CERTIFICATE

This is to certify that the thesis entitled **Indexing chemical fingerprints for efficient querying of molecular databases**, submitted by **Abhik Mondal**, to the Indian Institute of Technology, Madras, for the award of the degrees of **Bachelor of Technology** & **Master of Technology**, is a bona fide record of the research work carried out by him under my supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. Sayan Ranu
Research Guide
Assistant Professor
Dept. of Computer Science and Engineering
IIT-Madras, 600 036

Place: Chennai

Date:

ACKNOWLEDGEMENTS

ABSTRACT

Chem-informatics is a field dealing with chemistry and information science, with the primary motivation being the use of data mining, information retrieval and machine learning techniques to make predictions and inferences which can later be verified experimentally. Chemical Compounds are frequently represented as feature vectors where every feature represents the absence, presence or the frequency count of certain important substructures or other chemical properties. These feature vectors are known as ”Chemical Fingerprints” .

Fast database search is vital especially in drug discovery where the aim is identifying chemical compounds with high similarity to known drugs.In this work we are concerned with indexing methods of such datasets of chemical compounds to facilitate rapid range search querying on the database. We propose two techniques for the same. There is no loss of accuracy in any of the two methods when compared to a complete database linear scan .

The first technique uses an indexing technique based on the structure of the M-tree data structure. The standard similarity measure used for chemical fingerprints is the Tanimoto Similarity which satisfies triangle inequality. The M-tree structure helps us exploit this fact. The second technique we describe is based on an Inverted Indexing method which takes advantage of the sparsity and high dimensionality of a typical chemical fingerprint dataset. We go on to study the effectiveness of these techniques through various experiments.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

Introduction

Chem-informatics is largely concerned with the task of mining large chemical datasets. Typical applications of this include drug discovery, catalyst analysis and experiment predictions. Usually chemicals are represented as ”Chemical Fingerprints”.Chemical fingerprints are used for identification of chemicals. They are characteristics or distinctive patterns which help describe them. The comparison of chemical molecules is difficult but if we convert the molecules into bit-strings or into vector format it makes it easier to do a comprehensive comparison. A primary reason for using chemical fingerprints is scalability. We know that sub-graph isomorphism is an NP-complete problem making it difficult to find similarity between two molecules represented in a graphical format. When the data is represented in a vector string, techniques are more scalable in terms of size of query molecules and time to accomplish a similarity detection task.

Chemical fingerprints can be built based on structure properties like substructure features. In binary fingerprints each bit represents the presence or absence of specific chemical property like substructure presence. For example the bits could represent the count of individual chemical atoms like carbon, oxygen, hydrogen or the number of saturated or unsaturated aromatic carbon only, nitrogen containing or non-aromatic rings. The reason why substructure presence is is an important feature is because it can be used to reduce or filter out candidate mismatches in the sub-graph isomorphism test process. So given two bit-strings or vectors for binary chemical fingerprints we can intuitively say that the similarity of the two fingerprints is directly proportional to the number of shared bits.

These fingerprints/feature vectors are designed by domain experts and typical features could include information like number of occurrences of particular substructures, presence of molecules, bonds between the molecules, etc. Generally the dataset is high dimensional and also highly sparse which inherently makes searching and indexing such chemical data sets challenging.

For performing query operation on high dimensional data, it is generally observed that all the index-

ing techniques perform poorly, as compared to linear scan. This is mainly due to the curse of dimensionality . As dimensions increases the data points are scattered further away in space. In some techniques which use pivots to form clusters like the one we will use, it is difficult to form compact groups with a low cluster radius. Since we are unable to form compact representative groups well, more than often a range query requires a search through the entire database of chemical molecules with no chance of pruning occurring. Hence, in our work we have used "Linear scan" as our base line technique and have come up with methods to improve our performance. Our experiments will be evaluated against the linear scan technique. The results of the range search will be verified by comparing the range search result set against that obtained by a linear scan approach.

Various scoring schemes like Tversky, Pearson, Dice, and Kulczynski are available for comparing two given fingerprints [**? ?**].The most widely used and accepted similarity measure for binary fingerprints is the Tanimoto similarity which is equivalent to the Jaccard Similarity index defined as the size of the intersection of the sets divided by the size of union of the sets. Here the similarity measure would correspond to the number of bits which are 1 for both the sets of the fingerprints divided by the number of bits for which atleast one of the fingerprints is 1. In mathematical terms,for binary fingerprints X and Y ,where X_i is the i^th bit of X we define Tanimoto similarity as

$$T_s(X,Y) = \frac{\sum\limits_i X_i \wedge Y_i}{\sum\limits_i X_i \vee Y_i}$$

Here we can show that $1 - T_s$ is a proper distance metric preserving triangle inequality which will allow us to use index structures which exploit the property, like M-trees. One challenge which we face here is the extension of the said similarity measure /distance metric to non binary fingerprints. Non binary fingerprints are basically feature vectors where each feature value is a count and not necessary signifying absence or presence using bits

Similarity between chemical molecules plays an important role in various aspects of biology and chemistry like protein ligand docking, biological activity prediction,reaction site modelling and mainly drug discovery. Chemical or molecular similarity plays a pivotal role in modern techniques used to predict chemical compound properties, designing tailor-made chemicals with a predefined and desirable set of properties and most importantly in managing drug design studies by using large databases containing structures of available chemicals.

The first technique we used is that of M-tree [**?**] based indexing. In this technique we have proposed a novel pivot selection technique, which results in a efficient search time for range queries. We follow this up by embedding the data in to Euclidean space using Lipschitz embedding. We then perform a detailed analysis of the data, and we show that this analysis naturally leads us to inverted indexing. The large size of the data and the sparsity of the data are the primary challenges which we have addressed in this work.

CHAPTER 2

Related Work

In cheminfomatics the problem of fingerprint searching in a large database is well studied. In **?**] they present a new index-based search method called ChemDex (Chemical fingerprint inDexing) for speeding up the fingerprint database search. They propose a novel chain scoring scheme to calculate the Tanimoto (Jaccard) scores of the fingerprints using an early-termination strategy. The fingerprints are horizontally sorted by the total number of 1's they contain. A vertical sorting or rearrangement then takes place such that most of the 1's get pushed towards the left. After the shuffling vertical splits/slices are created in the database. A two tier index structure is created based on total number of 1's each data point has plus the number of 1's in each split. Since we have vertical slices in the database, when a query compound is given they propose a slice by slice fragment filtering.The bounds are calculated in the first slice and only the compounds which cross the threshold are checked for in the second slice. Since they are laterally traversing the slices and filtering after each slice the number of candidates are decreasing at each step hence reducing the time as compared to when we process the fingerprints in full. They come with a scoring technique such that the partial score at each slice is the upper bound for the final similarity score.

This field has also been well motivated in **?**]. As mentioned in the paper, fingerprint vectors are used to search large databases of small molecules, currently containing millions of entries, using various similarity measures, such as the Tanimoto or Tversky's measures and their variants. They derive simple bounds on these similarity measures and show how these bounds can be used to considerably reduce the subset of molecules that need to be searched.The paper proposes bounds for both single molecule as well as molecule molecule queries. The queries considered by the paper are based on threshold similarity cut-off with a query compound as well as top-k best set. The paper studied the speed-up achieved in query time against query size , query distribution, length of the chemical fingerprints, threshold cut-off for similarity and the total size of chemical database. They show through experiments that their approach achieves linear speed-ups in order of 1 or more magnitude for the fixed threshold query scenario while for top-k queries, they achieve sub-linear speed-ups in range of $O(D^{0.6})$ where D is size of database.

A different approach to the same approach can be seen in **?**] where to speed-up database searches,

they proposed to add to each binary fingerprint a short signature integer vector of length M. For a given fingerprint, the i-component of the signature vector counts the number of 1-bits in the fingerprint that fall on components congruent to i modulo M. Given two signatures, they show how one can rapidly compute a bound on the Jaccard-Tanimoto similarity measure of the two corresponding fingerprints, using the intersection bound. These signatures allow one to significantly prune the search space by discarding molecules associated with unfavourable bounds.

CHAPTER 3

Problem Formulation

3.1 Problem Statement

As the title suggests we are interested in fast indexing and searching of chemical fingerprints. We want to propose new indexing techniques which build on current indexing data structures and which will make searching for chemical compounds in the database faster. Our primary goal is to be able to perform queries on our compound database as fast as possible.

We are looking at accurate searching of similar compounds when given a query compound. The similarity of chemical fingerprints is established using the Min-Max distance which is the generalization of Tanimoto distance for non-binary datasets (explained in the next section). We are dealing with exact similarity match and not approximate similarity . We will be looking at different types of queries, namely the range, point and top-k queries .

Range queries are of the form , where given a fingerprint, say *f*, and a threshold similarity θ we want to find the set S of all fingerprints, such that

$$sim(f, g) \ > \ \theta \ \ \Rightarrow g \in S$$

Top k search queries would require us to find the top k most similar compounds to the query compound. Given a fingerprint, say f and a value *k*,we want to find the Set S of size k such that

$$sim(f, g) \geq sim(f, h) \ \ \forall g \in S, \ \forall h \notin S$$

. We haven't looked at these search queries in the experiments as of yet but plan to explore this area in the future.

Our aim as mentioned earlier is to come up with a novel indexing scheme which would make the aforementioned tasks faster. For range queries we would want to achieve sub-linear search time speeds.The challenge and novelty in our work comes from the fact that, unlike in **?**] and many other state of the art techniques developed in this domain, we are working on non-binary vector fingerprints.

3.2 Similarity measure

As mentioned in the earlier section the main challenge is the fact that we are dealing with non binary fingerprints which has seen very little work. Many of the papers mentioned in the related work suggest that their technique can be extended to non-binary fingerprints but fail to provide any experimental evaluation nor any details about the similarity measure used. We will be using a generalization of the Tanimoto similarity measure to incorporate non binary feature values. The similarity measure known as Min-Max similarity measure M_s between two fingerprints X, Y can be formulated as, where X_i is the i_{th} feature value of X

$$M_s(X,Y) = \frac{\sum_i min(X_i, Y_i)}{\sum_i max(X_i, Y_i)}$$

If the fingerprints are binary, this similarity measure reduces to the Tanimoto similarity. We need to ensure that the corresponding distance measure $M_d = 1 - M_s$is a metric. We can write the distance measure as:

$$M_d(X,Y) = 1 - M_s(X,Y) = 1 - \frac{\sum_i min(X_i, Y_i)}{\sum_i max(X_i, Y_i)}$$

$$M_d(X,Y) = \frac{\sum_i |X_i - Y_i|}{\sum_i max(X_i, Y_i)}$$

We can easily see that $M_d(X,X) = 0$ and that if $M_d(X,Y) = 0$ it implies $X_i = Y_i$ for all i, hence implying X=Y. The triangle inequality has been proved in **?**]

CHAPTER 4

Data

Before we get to the experiments, we will describe the data in this chapter. To get some more insight in to the behaviour of the indexing techniques that we are applying, we perform further analysis of the dat in the hope of getting some useful explanations. We want to ascertain that a typical fingerprint dataset is high dimensional and highly sparse almost follwing a power law sort of distribution with very few features occuring in almost all the points in the chemical database while majority of the features have a non-zero value in very few points. We want to extract patterns in the data which could be exploited for range search querying and point querying.

4.1 A Typical Fingerprint Dataset Analysis

We perform a detailed analysis of the data from **Dataset 1** to figure out the kind of indexing technique, which needs to be used to optimize our searching time. The statistics that were extracted from the data are as follows:

Number of data points	264016
Number of unique features is	785985
Maximum number of features in a data point is	1903
Minimum number of features in a data point is	7
Average number of features in a data point is	270.602966
Maximum number of data point with a feature is	259110
Minimum number of data point with a feature is	1
Average number of data point with a feature is	90
Maximum value of a feature	1870
Minimum value of a feature	1
Average value of a feature	1.142210
Maximum number of heavy-hitters	144
Minimum number of heavy-hitters	1
Average number of heavy-hitters	44.5

Table 4.1: Statistics of Data

From this we can observe that the data is highly sparse with only about 271 features, on a average, in a point, as opposed to the 785985 unique features. This high sparsity makes the data set an ideal candidate to perform inverted indexing. In inverted indexing, instead of indexing the data points we would index the features. This leads to a large index structure, but we gain on the speed of point query.

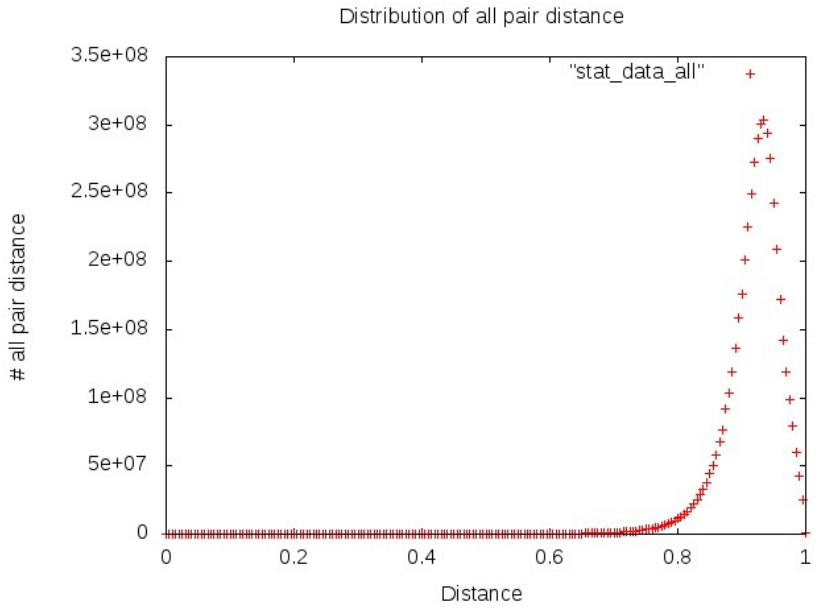


Figure 4.1: All pair distances

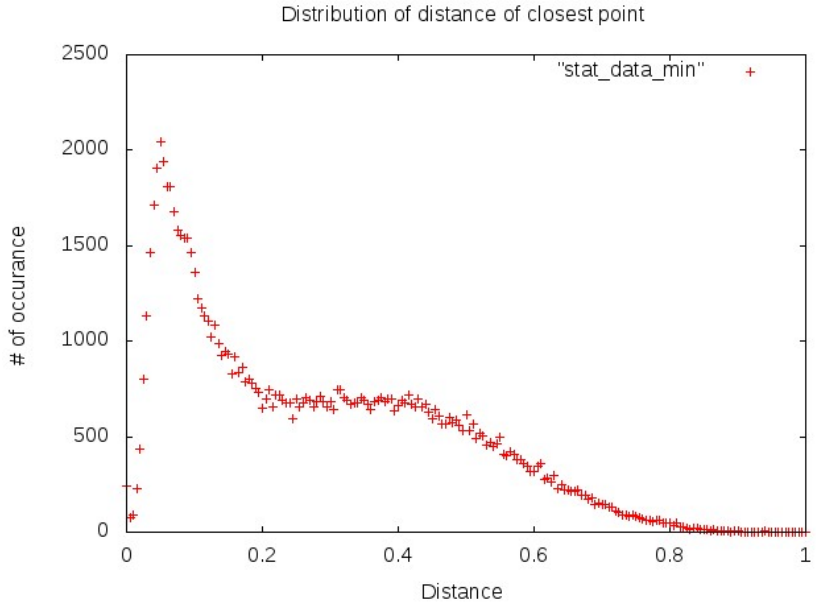


Figure 4.2: Distance of the closest point

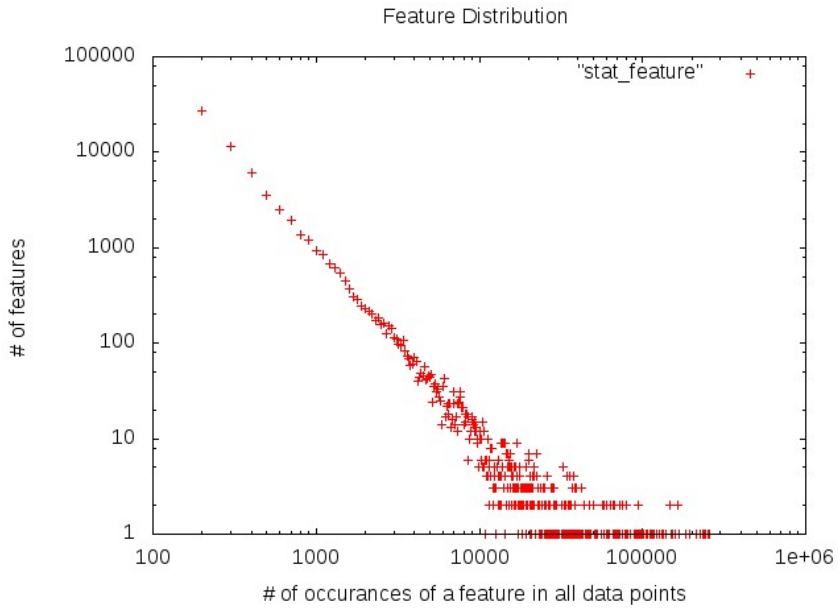


Figure 4.3: Feature distribution

4.2 Observations on the Data

What we can observe from the data are the following:

1. The being in high dimension, is very much spread out, and hence most of the points are equidistant from each other.

2. The closest point for most of the points is at distance much greater than 0.4. in fact only 0.007% of the points have their 1-NN within a distance of 0.4.

3. the distribution of features among the data points seem to follow a power law distribution (though we have not tried to regress the plot to a function) i.e., even though we have many features only a hand full of them are repeated in most of the points.

4. One of the most time consuming operations in our algorithms has been the recurring theme of finding the set of maximally separated points, which we called as pivots. It takes hours for the algorithm to converge. Given the distribution of all pairs distance, we cannot hope to improve it further, but we can propose an heuristic streaming algorithm to handle it efficiently.

   We can maintain a list of most frequently occurring farthest points. These points are not necessarily the points farthest from each other, but the points which are most frequent in the list farthest points, for each point. This is motivated from the frequency counting problem from streaming as shown in **?**].

5. We observed that the inverted index we proposed could not be easily generalised to range queries. This can be further observed from the fact that how skewed the data distribution is.

Given that only 0.007% of the 1-nn points have distance less that 0.4, we can never achieve a efficient pruning. And in addition the distribution of heavy-hitters (features with occurrence more that 50000) is also very high, We have on an average 44 heavy hitters in a data point. Hence, in the worst case, we will be forced to explore all the points.

6. The success of any embedding technique, especially Lipschitz, depends on the ability of the reference set to be representative of the entire data. in our case, given that the entire data is very widely spread, the number of reference set required to well represent the data, becomes very large. This is the reason why Lipschitz did not give the desired results.

CHAPTER 5

Experiments

In our experiments, we evaluated our indexing techniques on two real world datasets. We have compared our indexing techniques by comparing the running time with that of the state of the art technique . We are also concerned about the indexing time, especially for the M-tree index structure since we do not want our index procedure to run into hours.

The first dataset is the Swamidass dataset . The second dataset is from DUD, a directory of useful decoys for benchmarking virtual screening. DUD is provided by the Shoichet Laboratory in the Department of Pharmaceutical Chemistry at the University of California, San Francisco (UCSF). DUD is derived from ZINC, a database of commercially avaiilable compounds. To extarct fingerprints we used MOLPRINT2D, a molecular fingerprinting technique.

5.1 M-tree Indexing analysis

For these set of experiments the test bed used was a 4 Intel(R) Core(TM) i7-4770 CPU 3.40GHz with 8GB RAM. We have varied several parameters in the experiment and have tried to estimate them emperically.

For evaluation purposes we implemented a linear brute force scan to compute the range query and used that as a benchmark for the results. The result set obtained from our technique was compared with the linear brute scan answer set for verification purposes using the fingerprint id's. The query time and the indexing time is averaged over the 500 random query sample data points and the unit in ms per compound. We have varied the following parameters. We will just the paramters below:

1. No. of random pivots chosen at each step of the indexing process (p). *p* determines the number of nodes at every level of the index structure. The number of nodes at each level is equal to p+1.
2. Minimum limiting size of the outlier set allowed (o). When the outlier set size falls below this limit, we terminate the indexing process.
3. The range query distance threshold (t) is thse cutoff used for determining similarity.

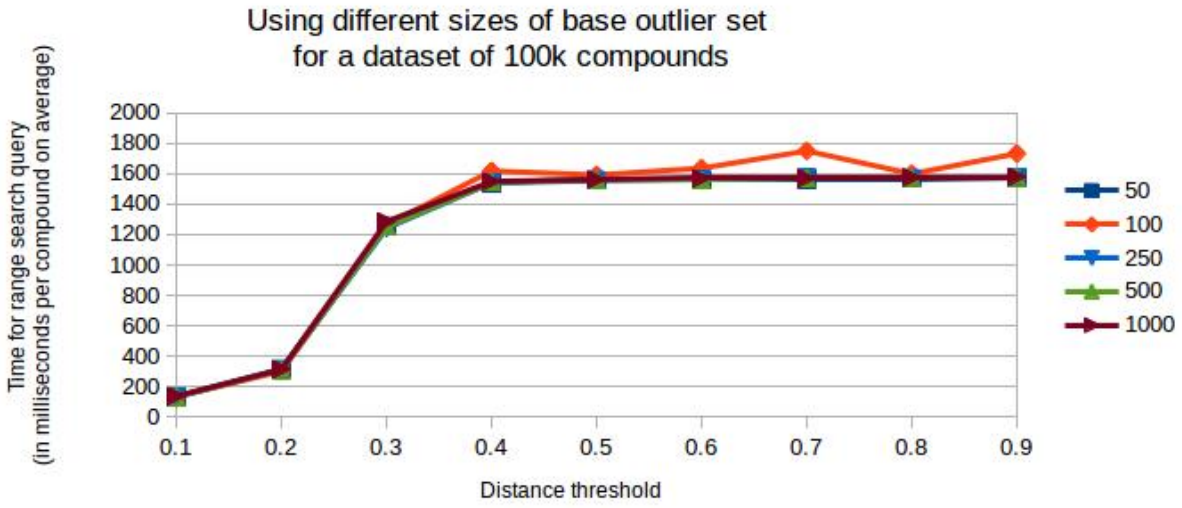


Figure 5.1: Average Range Search Query time versus Distance Threshold for various sizes of base outlier set

5.1.1 Limiting Outlier Set Size

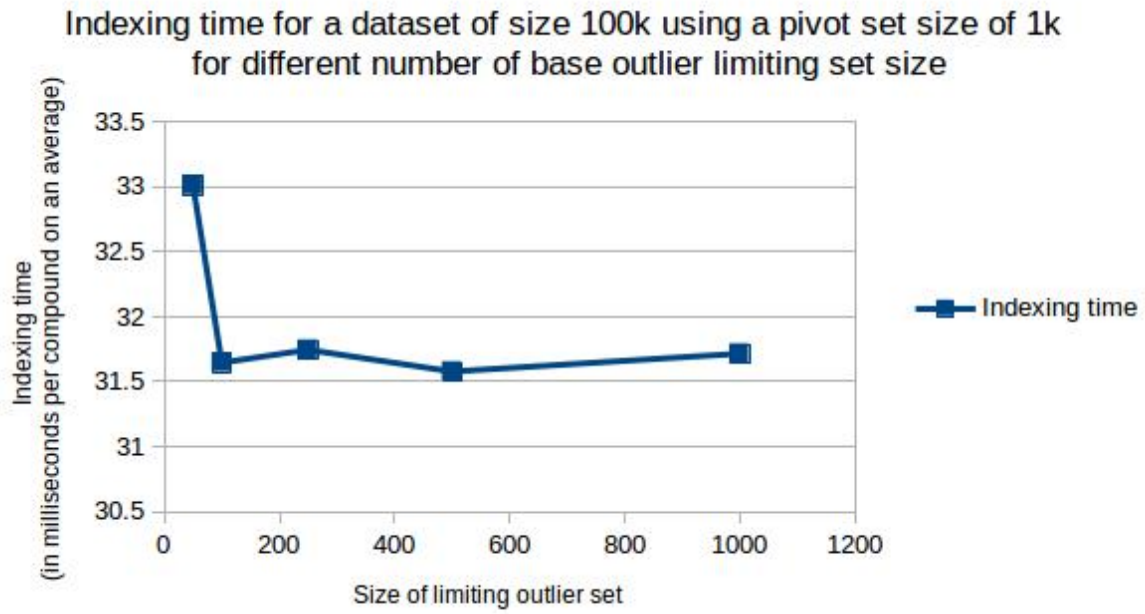


Figure 5.2: Indexing time versus different sizes of base limiting outlier set

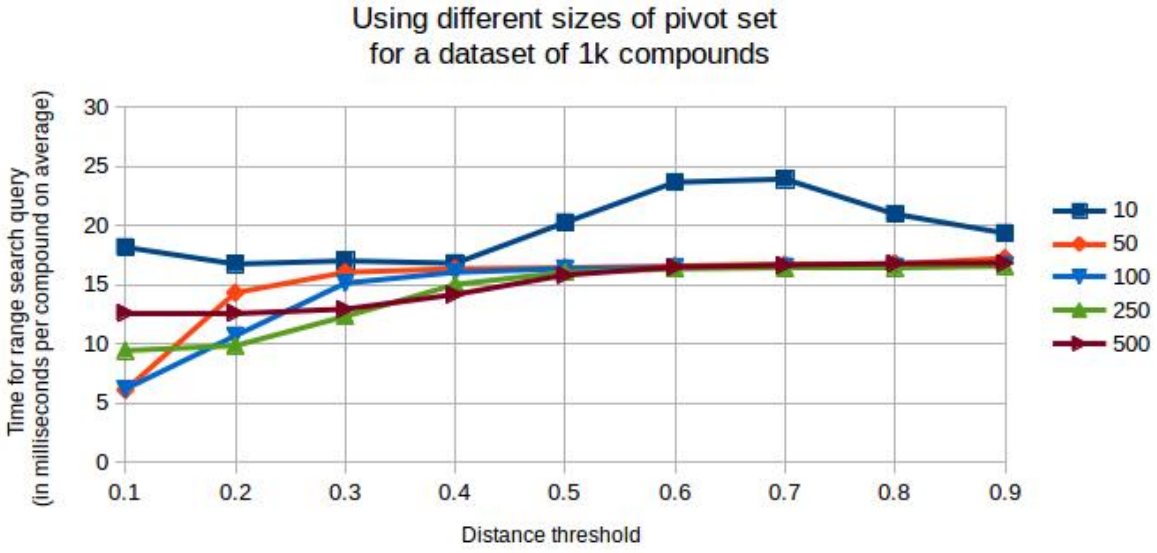


Figure 5.3: Average Range Search Query time versus Distance Threshold for various sizes of pivot set - 1k database

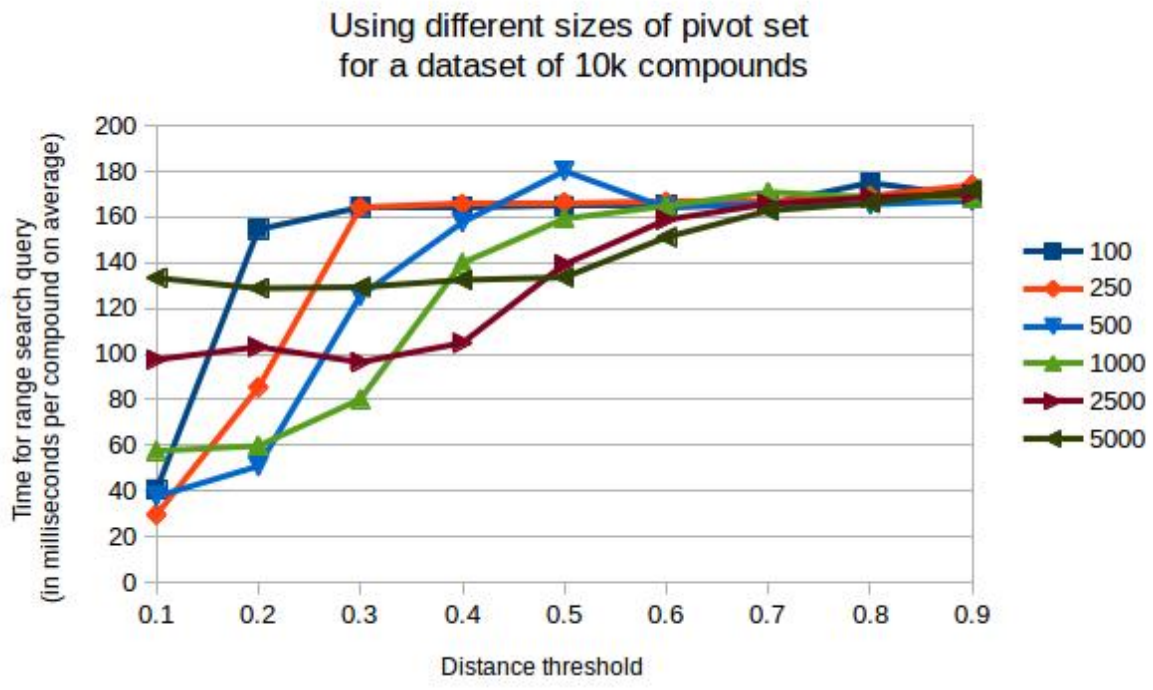


Figure 5.4: Average Range Search Query time versus Distance Threshold for various sizes of pivot set - 10k database

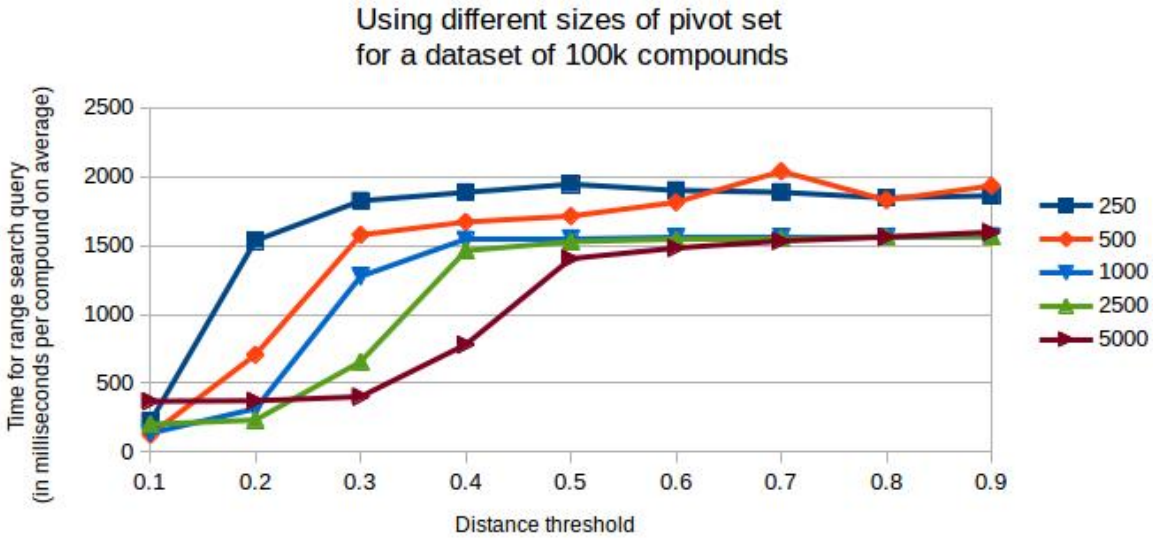


Figure 5.5: Average Range Search Query time versus Distance Threshold for various sizes of pivot set - 100k database

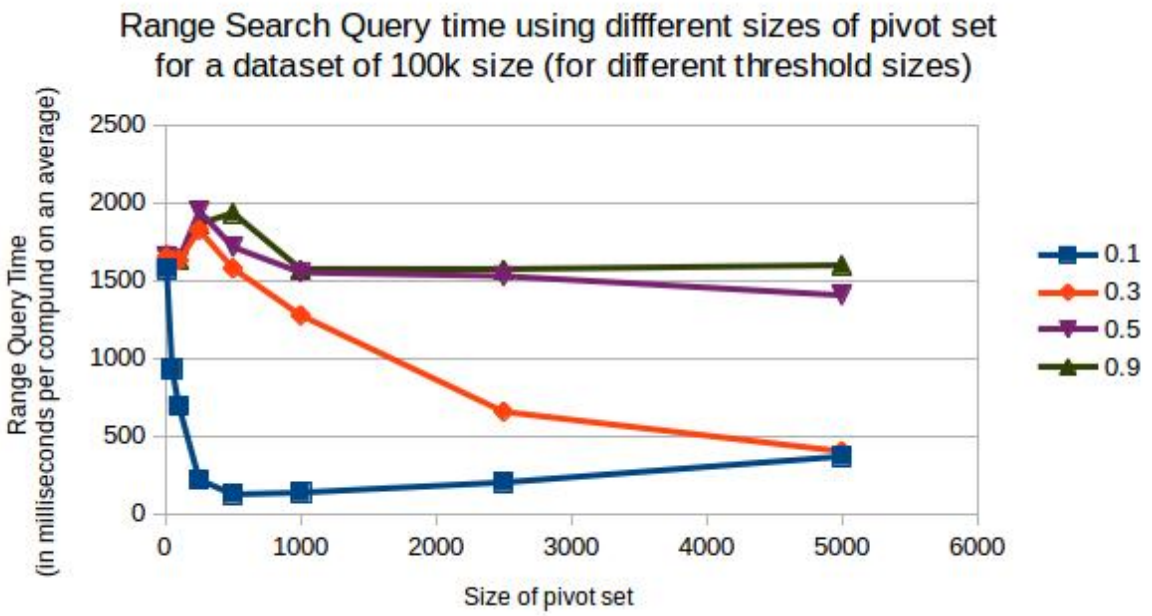


Figure 5.6: Average Range Search Query time versus various sizes of pivot set for different theshold distances

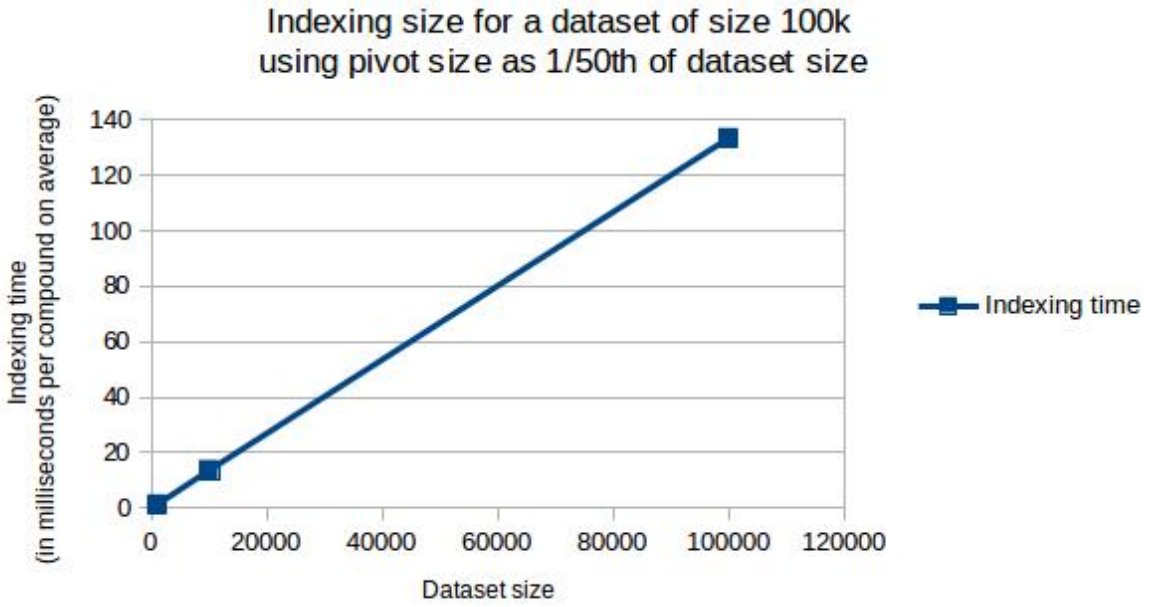


Figure 5.7: Indexing time versus dataset size

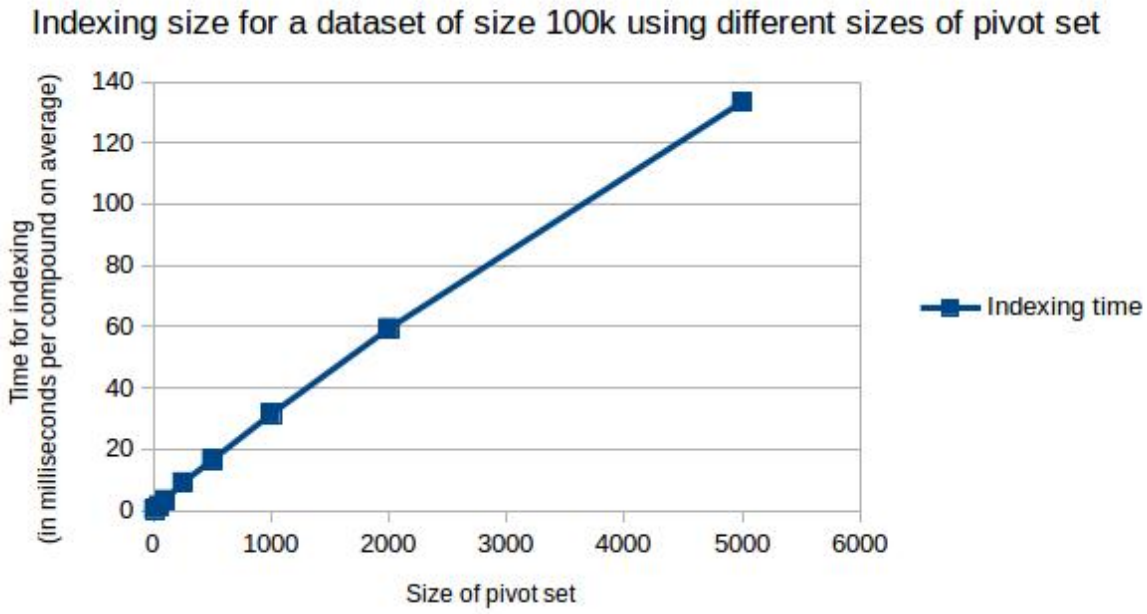


Figure 5.8: Indexing time versus different sizes of pivot set size

REFERENCES