**Nearest Keyword Set Search in Multi-Dimensional Datasets**

**ABSTRACT:**

Keyword-based search in text-rich multi-dimensional datasets facilitates many novel applications and tools. In this paper, we consider objects that are tagged with keywords and are embedded in a vector space. For these datasets, we study queries that ask for the tightest groups of points satisfying a given set of keywords. We propose a novel method called ProMiSH (Projection and Multi Scale Hashing) that uses random projection and hash-based index structures, and achieves high scalability and speedup. We present an exact and an approximate version of the algorithm. Our experimental results on real and synthetic datasets show that ProMiSH has up to 60 times of speedup over state-of-the-art tree-based techniques.

**EXISTING SYSTEM:**

* Location-specific keyword queries on the web and in the GIS systems were earlier answered using a combination of R-Tree and inverted index.
* Felipe et al. developed IR2-Tree to rank objects from spatial datasets based on a combination of their distances to the query locations and the relevance of their text descriptions to the query keywords.
* Cong et al. integrated R-tree and inverted file to answer a query similar to Felipe et al. using a different ranking function.

**DISADVANTAGES OF EXISTING SYSTEM:**

* These techniques do not provide concrete guidelines on how to enable efficient processing for the type of queries where query coordinates are missing.
* In multi-dimensional spaces, it is difficult for users to provide meaningful coordinates, and our work deals with another type of queries where users can only provide keywords as input.
* Without query coordinates, it is difficult to adapt existing techniques to our problem.
* Note that a simple reduction that treats the coordinates of each data point as possible query coordinates suffers poor scalability.

**PROPOSED SYSTEM:**

* In this paper, we consider multi-dimensional datasets where each data point has a set of keywords. The presence of keywords in feature space allows for the development of new tools to query and explore these multi-dimensional datasets.
* In this paper, we study nearest keyword set (referred to as NKS) queries on text-rich multi-dimensional datasets. An NKS query is a set of user-provided keywords, and the result of the query may include k sets of data points each of which contains all the query keywords and forms one of the top-k tightest cluster in the multi-dimensional space.
* In this paper, we propose ProMiSH (short for Projection and Multi-Scale Hashing) to enable fast processing for NKS queries. In particular, we develop an exact ProMiSH (referred to as ProMiSH-E) that always retrieves the optimal top-k results, and an approximate ProMiSH (referred to as ProMiSH-A) that is more efficient in terms of time and space, and is able to obtain near-optimal results in practice.
* ProMiSH-E uses a set of hashtables and inverted indexes to perform a localized search.

**ADVANTAGES OF PROPOSED SYSTEM:**

* Better time and space efficiency.
* A novel multi-scale index for exact and approximate NKS query processing.
* It’s an efficient search algorithms that work with the multi-scale indexes for fast query processing.
* We conduct extensive experimental studies to demonstrate the performance of the proposed techniques.

**SYSTEM ARCHITECTURE:**



**MODULES:**

* The Index Structure For Exact Search(ProMiSH-E)
* The Exact Search Algorithm
* Optimization Techniques
* The Approximate Algorithm (ProMiSH-A)

**MODULES DESCRIPTION:**

**The Index Structure for Exact Search (ProMiSH-E):**

* In This Project we start with the index for exact ProMiSH (ProMiSH-E). This index consists of two main components.
* **Inverted Index Ikp**: The first component is an inverted index referred to as Ikp. In Ikp, we treat keywords as keys, and each keyword points to a set of data points that are associated with the keyword. Let D be a set of data points and V be a dictionary that contains all the keywords appearing in D. We build Ikp for D as follows. (1) For each ,we create a key entry in I kp, and this key entry points to a set of data points (i.e., a set includes all data points in D that contain keyword v). (2) We repeat (1) until all the keywords in V are processed.
* **Hashtable-Inverted Index Pairs HI**: The second component consists of multiple hashtables and inverted indexes referred to as HI. HI is controlled by three parameters: (1) (Index level) L, (2) (Number of random unit vectors) m, and (3) (hashtable size) B. All the three parameters are non-negative integers. These three parameters control the construction of HI.

**The Exact Search Algorithm:**

* We present the search algorithms in ProMiSH-E that finds top-k results for NKS queries. First, we introduce two lemmas that guarantee ProMiSH-E always retrieves the optimal top-k results.
* We project all the data points in D on a unit random vector and partition the projected values into overlapping bins of bin-width. If we perform a search in each of the bins independently, that the top-1 result of query Q will be found in one of the bins. ProMiSH-E explores each selected bucket using an efficient pruning based technique to generate results. ProMiSH-E terminates after exploring HI structure at the smallest index level s such that all the top-k results have been found. The efficiency of ProMiSH-E highly depends on an efficient search algorithm that finds top-k results from a subset of data points.

**Optimization Techniques**

* An algorithm for finding top-k tightest clusters in a subset of points. A subset is obtained from a hashtable bucket Points in the subset are grouped based on the query keywords. Then, all the promising candidates are explored by a multi-way distance join of these groups. The join uses rk, the diameter of the kth result obtained so far by ProMiSH-E, as the distance threshold.
* A suitable ordering of the groups leads to an efficient candidate exploration by a multi-way distance join. We first perform a pairwise inner joins of the groups with distance threshold rk. In inner join, a pair of points from two groups are joined only if the distance between them is at most rk.
* We propose a greedy approach to find the ordering of groups. The weight of an edge is the count of point pairs obtained by an inner join of
the corresponding groups. The greedy method starts by selecting an edge having the least weight. If there are multiple edges with the same weight, then an edge is selected at random and we perform a multi-way distance join of the groups by nested loops.

**The Approximate Algorithm (ProMiSH-A):**

* The approximate version of ProMiSH referred to as ProMiSH-A. We start with the algorithm description of ProMiSH-A, and then analyze its approximation quality.
* ProMiSH-A is more time and space efficient than ProMiSH-E, and is able to obtain near-optimal results in practice. The index structure and the search method of ProMiSH-A are similar to ProMiSH-E.
* The index structure of ProMiSH-A differs from ProMiSH-E in the way of partitioning projection space of random unit vectors. ProMiSH-A partitions projection space into non-overlapping bins of equal width, unlike ProMiSH-E which partitions projection space into overlapping bins. The search algorithm in ProMiSH-A differs from ProMiSH-E in the termination condition. ProMiSH-A checks for a termination condition after fully exploring a hashtable at a given index level: It terminates if it has k entries with nonempty data point sets in its priority queue PQ.

**SYSTEM REQUIREMENTS:**

**HARDWARE REQUIREMENTS:**

* System : Pentium Dual Core.
* Hard Disk : 120 GB.
* Monitor : 15’’ LED
* Input Devices : Keyboard, Mouse
* Ram : 1GB.

**SOFTWARE REQUIREMENTS:**

* Operating system : Windows 7.
* Coding Language : JAVA/J2EE
* Tool : Netbeans 7.2.1
* Database : MYSQL

**REFERENCE:**

Vishwakarma Singh, Bo Zong, and Ambuj K. Singh, “Nearest Keyword Set Search in Multi-Dimensional Datasets”, **IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 28, NO. 3, MARCH 2016.**