Applied Mathematics and Computation 328 (2018) 203–223

Contents lists available at ScienceDirect

Applied Mathematics and Computation

journal homepage: www.elsevier.com/locate/amc

Characterizing the optimal pivots for efficient similarity searches in vector space databases with Minkowski distances

Carolina Gómez-Tostón*, Manuel Barrena, Álvaro Cortés

University of Extremadura, Spain Ingenieering in Computer and Telematic Systems Avda de la Universidad, s/n 10001 Cáceres, Spain

ARTICLE INFO

Keywords:
Information search and retrieval
Indexing methods
Information filtering
Similarity search
Metric access methods
Multimedia databases

ABSTRACT

Pivot-based retrieval algorithms are commonly used to solve similarity queries in a number of application domains, such as multimedia retrieval, biomedical databases, time series and computer vision. The query performances of pivot-based index algorithms can be significantly improved by properly choosing the set of pivots that is able to narrow down the database elements to only those relevant to a query. While many other approaches in the literature rely on empirical studies or intuitive observations and assumptions to achieve effective pivot strategies, this paper addresses the problem by using a formal mathematical approach. We conclude in our study that the optimal set of pivots in vector databases with L^p metrics is a set of uniformly distributed points on the surface of an n-sphere defined by these metrics. To make the study mathematically tractable, a uniform distribution of data in the database is assumed, allowing us to outline the problem from a purely geometrical point of view. Then, we present experimental results demonstrating the usefulness of our characterization when applied to real databases in the $(\mathbb{R}^n, L^p)$ metric space. Our technique is shown to outperform comparable techniques in the literature. However, we do not propose a new pivot-selection technique but rather experiments that are designed exclusively to show the usefulness of such a characterization.

1. Introduction

Searching is a fundamental problem in computer science. Exact-match retrieval, typical for traditional databases, is neither feasible nor meaningful for many current data types that lack a defined structure, such as images, videos, and sound files. For these types of data, similarity search is a more frequently used search paradigm. Similarity search, given a particular object called the query object, addresses the problem of finding the objects in a database that are more similar to the query object. Because similarity is a subjective concept that is often difficult to quantify, it is usually measured by means of a distance function that in some way captures the concept of dissimilarity between objects.

The similarity search topic has attracted the interest of the database community due to the increase in the volume of unstructured data. In fact, it has become a fundamental computational task in a variety of application areas, such as multimedia information retrieval, data mining, pattern recognition, machine learning, computer vision, data compression, statistical data analysis, biomedical and chemical databases. In these researching fields, the data are represented as a collection of complex objects, like graphs or vectors. On the one hand, an example of graph representation is molecular information in

* Corresponding author.
E-mail addresses: cgomezt@unex.es (C. Gómez-Tostón), mbarrena@unex.es (M. Barrena), alvarocf@unex.es (Á. Cortés).

https://doi.org/10.1016/j.amc.2018.01.028
0096-3003/

chemistry, which is usually represented by picturing molecules as labeled graphs [1]. In general, graph similarity has been a very fruitful area and graph matching techniques have found ample applications in various scientific disciplines, giving way to a large variety of distance functions [2]. On the other hand, an example of vector representation is content-based image retrieval applications, where images are represented by feature vectors. A feature vector describes particular properties of the objects of interest for an application, as the color, texture, shape or other local properties of the image itself. This paper focuses on vector representations.

Once an object is represented by its corresponding feature vectors, the dissimilarity among the original objects can be measured using a distance function defined on the feature vector space, where the similarity between two objects increases as their corresponding feature vectors become closer to each other in the metric space. The user chooses the metric to be used based on the nature of the features. For instance, for feature histograms, the most prominent similarity measures are the L^p norm-based ones [3], although measures such as the quadratic form distance (QFD) [4,5] and others [3,6] can be also used.

Applications based on similarity search need to compute a large number of distances between the query object and the objects in the database. In general, computing distances is an expensive operation; thus, reducing the number of distance computations in the execution of a query has become a subject of interest to researchers [7–10]. To reduce the number of distance computations carried out during a search, several distance-based indexing algorithms have emerged. They basically try to prune the search space, avoiding a full scan of a database by rejecting objects that are definitely not part of the actual outcome of the query. The greater the number of objects rejected, the more efficient the algorithm. This is the key point of distance-based indexing algorithms. According to the classification of Chávez et al. [7], distance-based indexing algorithms are typically divided into two main classes: partitioning-based and pivot-based algorithms.

Partitioning-based algorithms split up the space of objects into a number of regions in an attempt to prune some of them during the search process. Two basic partition schemes can be identified [11]: (i) Ball partitioning (BST [12], VT [13], MT [14]), and (ii) Generalized hyperplane partitioning (GHT [11]). A hybrid approach is SAT [15], and another method based on partitions that does not fit with any of these schemes is GNAT [16].

Pivot-based algorithms choose a set of k pivots $P = \{p_1, \ldots, p_k\}$ that act as reference points during a search. Distances from all objects in the database to the pivots are calculated in advance and stored in an index. Indexed distances are then used to prune database objects (by using the triangular inequality) without computing their actual distances from the query. There are many algorithms that are based on pivots, such as BKT [12], FQT and FHFQT [17], FQA [18], VPT [19] and its variants [20], and AESA [21]. The latter is one of the most well-known methods and has been widely used as a baseline for performance measurements. Some variants of AESA are LAESA [22] and enhanced versions such as tree AESA [23], iAESA [24] and PiAESA [25].

The efficiency of pivot-based algorithms lies in the number of selected pivots k and their placement. It is well known that the way in which pivots are selected for searching in metric spaces has a strong influence on the search performance [7,22,26]. Hence, picking effective pivots [27–30] and fixing their optimal numbers [31,32] is a subject of research. However, finding the optimal set of pivots that minimizes the cost of a typical query still remains an open issue. In fact, no specific pivot selection technique has been shown to be optimal in all cases [25].

1.1. Paper scope, contributions and organization

This paper focuses on the efficiency of pivot selections from an analytical point of view. In this paper, we present a formal attempt to characterize the optimal set of pivots in vector databases with L^p or Minkowski metrics, $(\mathbb{R}^n, L^p)$. Our characterization demonstrates that optimal pivots are uniformly distributed over the surface of an n-ball centered at the geometric center of the database and with a radius that expands as the number of pivots to be used or the range query radius increases, possibly exceeding the database boundaries. Such an n-ball is defined by the L^p metric, and its surface is commonly known as an $n-1$-sphere; however, as a convenient abuse of terminology, we will refer to the $n-1$-sphere simply as the n-sphere throughout this paper. In analytic geometry, an n-sphere is the locus of all the points that are L^p-equidistant to another one and is called the center. Fig. A.1 shows different unit 2-spheres in different L^p metrics.

The optimal pivot characterization is developed gradually. First, we conduct a geometrical analysis based on a mathematical modeling of the pivot-filtering problem in $(\mathbb{R}^2, L^2)$. To make the analysis tractable, uniform conditions for the database are imposed. The result of this analysis proves that the optimal pivots lie uniformly distributed in a circle. Second, we establish the characterization for the n-dimensional case $(\mathbb{R}^n, L^2)$. Finally, we extend the result to the generic L^p metric space $(\mathbb{R}^n, L^p)$. Although our characterization is based on a uniform data distribution, experimental work conducted on real databases demonstrates that search pivot-based algorithms that adopt such a characterization outperform the most relevant pivot-selection techniques. Nevertheless, introducing a new pivot selection algorithm is out of the scope of this paper, and the experiments on real databases are designed solely to support the usefulness of our characterization.

The remainder of this paper is organized as follows. The mathematical support of our study on pivot filtering is introduced in Section 2. Section 3 presents the body of our analysis, where we formally define a characterization of optimal pivots in Euclidean metric spaces $(\mathbb{R}^n, L^2)$, that is, a pivot set minimizing the query cost. Section 4 generalizes this characterization to general Minkowski metric spaces $(\mathbb{R}^n, L^p)$. In Section 5, we introduce the most relevant works related to the search efficiency of pivot-selection techniques. In Section 6, the experimental work on real databases is presented, and further discussions regarding our contributions are given. Final conclusions and future work are presented in Section 7.