Information Systems 43 (2014) 1–19

Contents lists available at ScienceDirect

Information Systems

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

On indexing metric spaces using cut-regions☆

Jakub Lokoč*, Juraj Moško, Přemysl Čech, Tomáš Skopal

Charles University in Prague, Faculty of Mathematics and Physics, SIRET Research Group, Malostranské nám., 11800 Prague, Czech Republic[1]

ARTICLE INFO

Article history:
Received 10 July 2013
Received in revised form
21 January 2014
Accepted 24 January 2014
Recommended by: F. Korn
Available online 1 February 2014

Keywords:
Indexing methods
Multimedia databases
Metric access methods
PM-tree
M-Index
List of Clusters

ABSTRACT

After two decades of research, the techniques for efficient similarity search in metric spaces have combined virtually all the available tricks resulting in many structural index designs. As the representative state-of-the-art metric access methods (also called metric indexes) that vary in the usage of filtering rules and in structural designs, we could mention the M-tree, the M-Index and the List of Clusters, to name a few. In this paper, we present the concept of *cut-regions* that could heavily improve the performance of metric indexes that were originally designed to employ simple ball-regions. We show that the shape of cut-regions is far more compact than that of ball-regions, yet preserving simple and concise representation. We present three re-designed metric indexes originating from the above-mentioned ones but utilizing cut-regions instead of ball-regions. We show that cut-regions can be fully utilized in the index structure, positively affecting not only query processing but also the index construction. In the experiments we show that the re-designed metric indexes significantly outperform their original versions.

1. Introduction

Although there have been many *metric access methods* (or metric indexes) [1–4] developed in the past decades, there still emerge new metric access method (MAM) designs and other approaches addressing the problem of efficient processing of similarity queries. In the last years we observe a trend towards even more complex MAM structures represented by, e.g., the M-Index [5], the D-file [6], the pivot table (and all its variants) [7], the permutation indexes [8], and others that are often based on transformation of the metric space model into another geometric model. The “good old” indexing structures that directly partition the metric space, e.g., the M-tree [9], the (m)vp-tree, the GNAT [10,11], etc., are often outperformed by the new MAMs. From this perspective, it might seem that the MAMs relying on direct hierarchical partitioning of the metric space bring an unnecessary overhead and so they should be abandoned. However, although the partitioning-based MAMs exhibit worse performance in traditional queries, such as the range query or the k nearest neighbor query [6], for modern retrieval modalities the compact hierarchies of metric regions could perform much better. For instance, various iterative queries within the *multimedia exploration* area [12] could benefit from the native hierarchy of metric regions where a continuous traversal in the metric space is required. Another application proving the benefits of metric partitioning is demonstrated by the M-Index that efficiently combines partitioning with the iDistance [13] mapping approach. Here a compact hierarchy is crucial if the M-Index is distributed among many machines [14].

In this paper, we define a new formalism for construction of compact metric regions – the *cut-regions*. The formalism enables to simplify the adaptation of complex MAM algorithms using ball-regions to employ the cut-regions instead. In particular, we show how the formalism can be used for re-definition of the PM-tree structure and

☆ This paper is an extended version of a previous paper by Lokoč et al., [31].

* Corresponding author.

E-mail addresses: lokoc@ksi.mff.cuni.cz (J. Lokoč), mosko@ksi.mff.cuni.cz (J. Moško), Premekcech@seznam.cz (P. Čech), skopal@ksi.mff.cuni.cz (T. Skopal).

[1] http://www.siret.cz.

0306-4379/$ - see front matter © 2014 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.is.2014.01.007

its construction algorithms. Based on the cut-regions we introduce new PM-tree construction algorithm that leads to more compact PM-tree hierarchies (and so faster similarity search). Note that compact hierarchy of metric regions is not only beneficial for efficiency of traditional queries (range or k NN), but it can also better serve as a hierarchy of clusters that can be used in exploration queries, data mining, and other tasks. We also implement cut-regions to other two state-of-the-art MAMs – the M-Index and the List of Clusters.

1.1. Paper contributions

The paper contributions can be summarized into four main points:

- The new cut-region formalism that is suitable for simplified description of compact metric regions. Cut-regions can be utilized in new or existing metric indexing structures and algorithms, as demonstrated on the PM-tree.
- New cheap dynamic construction techniques for the PM-tree that can compete with expensive strategies of the original PM-tree (e.g., multi-way leaf selection).
- Adaptation of M-Index and List of Clusters to operate with cut-regions.
- Thorough experimental evaluation also including comparison with the state-of-the-art MAMs.

The rest of the paper is organized as follows. For readers not familiar with the metric search approach we provide a quick overview in Section 2. As the cut-region is the key concept used in this paper, we precisely define the cut-regions and basic operations on them in the following Section 3. Then, we redefine the PM-tree index using cut-regions in Section 4. In Section 5, we present new dynamic PM-tree construction techniques, while in Section 6 we describe other MAMs that can benefit from cut-regions. We provide thorough experimental evaluation in Section 7 and, finally, we conclude the paper and sum up its main contributions in Section 8.

2. Similarity search essentials

Having a collection of complex unstructured objects (like multimedia documents, texts, time series, 3D models, etc.), the search in such collection can hardly be based on traditional query models that assume the user is familiar with an explicit structure of the data (e.g., relational schema used by SQL). Instead, the unstructured objects have to be transformed into structured *feature descriptors* (or descriptor objects)[2] by means of a feature extraction procedure. In this step a similarity model is established, consisting of a universe $\mathcal{D}$ of feature descriptors and a function δ for measuring dissimilarity/distance of any pair of feature descriptors. Using a similarity model, the collection of descriptors can be searched using the query-by-example paradigm (e.g., return the 5 most similar images to my image of a dog).

2.1. Similarity search

Similarity queries are an intuitive way of how to express search intent on some objects in a given domain. Usually, we want to find the k most similar objects to a given query object q or just find all objects within a distance r from the query object q. These types of queries are called the k nearest neighbor (k NN) query and the range query, respectively. Although different query types were designed and utilized for image retrieval problems (Section 1.4 in [2]), these two are the most common ones.

Definition 1 (*Range query*). Let $q \in \mathcal{D}$ be a query object and $r \in \mathbb{R}_0^+$ be a query radius (or a distance threshold). Range query is defined as $R(q,r) = \{o \in X, \delta(o,q) \le r\}$, where δ is a distance function on domain $\mathcal{D}$ and $X \subseteq \mathcal{D}$ is a dataset to be searched.

Definition 2 (*k nearest neighbor query*). Let $q \in \mathcal{D}$ be a query object and $k \in \mathbb{N}$ be a number of requested nearest neighbors. The k nearest neighbor query is defined as $k-NN(q) = \{R \subseteq X, |R| = k \wedge \forall x \in R, y \in X-R : \delta(q,x) \le \delta(q,y)\}$, where δ is a distance function on domain $\mathcal{D}$ and $X \subseteq \mathcal{D}$. If multiple such sets R exist, one is chosen arbitrarily.

2.2. Metric space model for similarity search

In order to search the data collections efficiently (quickly), the similarity function δ in the similarity model is often restricted to be a metric distance, hence obtaining the *metric space model* (see Definition 3). The properties of metric space enable to construct cheap lower bounds of the original (computationally expensive) similarity function which, in turn, are the basis for efficient similarity search. For more details on construction of lower bounds and on principles of metric indexing in general we refer the reader to a monograph [2] or survey [1].

Definition 3 (*Metric space*). Let $\mathcal{D}$ be a domain of feature descriptors, δ: $\mathcal{D} \times \mathcal{D} \mapsto \mathbb{R}$ a pairwise distance function on $\mathcal{D}$. Then $\mathcal{M} = (\mathcal{D}, \delta)$ is called a metric space, if the following postulates hold $\forall x,y,z \in \mathcal{D}$:

(p1) $\delta(x,x) = 0$	reflexivity
(p2) $x \neq y \Rightarrow \delta(x,y) > 0$	positiveness
(p3) $\delta(x,y) = \delta(y,x)$	symmetry
(p4) $\delta(x,z) \le \delta(x,y) + \delta(y,z)$	triangle inequality

Despite the restriction of the similarity function to a metric distance the metric space model still remains extensible enough to fit the needs of domain experts (i.e., practitioners from various domains managing large data volumes). The metric space approach enables to utilize not only vector spaces for modeling data, but also nonvectorial descriptor types, like strings, sets, time series, etc., and appropriate nonvectorial distance

[2] In the rest of the text the term *object* is used in the meaning of descriptor object/feature descriptor.