COPYRIGHT NOTICE

DOI: `10.1007/978-3-642-41062-8_9`.

Principal Directions-Based Pivot Placement

Fabrizio Angiulli and Fabio Fassetti

DIMES Department – University of Calabria, Rende, Italy
{f.angiulli, f.fassetti}@dimes.unical.it

Abstract. Determining a good sets of pivots is a challenging task for metric space indexing. Several techniques to select pivots from the data to be indexed have been introduced in the literature. In this paper, we propose a pivot placement strategy which exploits the natural data orientation in order to select space points which achieve a good alignment with the whole data to be indexed. Comparison with existing methods substantiates the effectiveness of the approach.

1 Introduction

The *similarity search* in metric spaces [7, 12, 14] is a fundamental task in a huge set of fields. In particular, *range queries*, which are of interest here, take as input the query object q and a radius R, and return all the objects of the dataset lying within distance R from q.

One of the main techniques for *indexing* objects from a metric space, is the *pivoting* based one [13, 10, 6, 7, 14, 2]. In such approaches, the idea is to select a certain number of objects, called *pivots*. Due to the reverse triangle inequality, given two generic objects x and y, their distance cannot be smaller than the distance between x and p minus the distance between y and p, for any other object p. Hence, in order to answer to a range query, this lower bound can be exploited to discard those objects which do not lie within distance R from the query object q, namely whose lower bound is greater than R. By sketchily summarizing, at indexing time all the pairwise distances among the objects of the dataset and the pivots are stored in the index. At query time, first of all, the distances between the query object q and all the pivots are computed. Then, a *candidate selection phase* follows, which is accomplished by selecting those objects which are not discarded by exploiting the lower bounds computed through the pivots. The true neighbors of q are eventually retrieved by a *filtering phase* consisting in computing the actual distances among q and each candidate object.

In the best case, the objects returned by the candidate selection phase coincide with the query answer. However, minimizing the number of the spurious objects is a hard challenge. Usually, the greater the number of pivots, the smaller the number of spurious objects in the candidate set. By keeping fixed the number of pivots, it is known that a clever selection of pivots can drastically improve index performances. Given a dataset and an integer k, the *pivot selection problem* accounts for selecting a good set of k objects to be employed as pivots [1]. Normally, pivots are selected among the set of objects to be indexed. In some

scenarios this may be the unique option, since it can be difficult to figure what a reasonable object out of those actually belonging to the data at hand can be. This is precisely the assumption made by all pivot selection techniques so far introduced in the literature [4, 11, 5].

Authors of [4] propose three different techniques, detailed in the following. According to the *Selection of N Random Groups* strategy, N groups each consisting in k pivots are randomly chosen from the dataset. For each group, the quality of the set of pivots is measured and the group scoring the maximum value of quality is returned. As for the *Incremental Selection* strategy, the idea is to randomly select N objects from the dataset. Then, the object (among the N selected ones) that alone scores the maximum value of quality is chosen as the first pivot p_1. The second pivot is the object p_2 such that maximizes the quality of the set of pivots $\{p_1, p_2\}$ and so on until k pivots are chosen. Finally, the *Local Optimum Selection* strategy consists in selecting an initial set of k pivots and a set of A pairs of objects. For N' times a so-called *victim* in the set of pivots is singled out and replaced by a better pivots chosen in a sample of X dataset objects. In order to individuate the victim, it is built a $A \times k$ matrix M, where $M(i, j)$ is the distance between the objects of the i^{th} pair computed through the pivot p_j. For each row, the maximum (d_{M}) and the second maximum (d_{M2}) values are computed. The *contribution* of each pivot p_j is computed as the sum over the A pairs of the difference between d_{M} and d_{M2}, if d_{M} is achieved in the column j and 0 otherwise. The *victim* is the pivot scoring the lowest contribution.

In [11] the *Sparse Spatial Selection* technique is proposed. The approach is based on the idea that if the pivots are well-distributed in the metric space they are able to discard more objects. The set of pivots is initially a singleton consisting in the first dataset object. Then, for each dataset object, this is chosen as a new pivot if and only if its distance from any pivot currently in the set is greater than $M\alpha$ where M is the maximum distance between any pair of dataset objects and α is a parameter whose empirically proven good value is 0.4. Such a method has next been extended in [5] where not only new pivots are added to the current set, but it is also checked if some pivot has become redundant and, in such a case, it is replaced by a better one. This latter task is accomplished by a estimating the contribution of an object in a fashion similar to the *Local Optimum Selection* approach.

As a main contribution, in this work we take a different perspective by considering the whole object domain in order to single out pivots, that is pivots can be objects that do not belong to the current dataset. We call this instance of the problem as *pivot placement problem*. Specifically, we consider here as reasonable scenario the case in which the object domain coincides with the d-dimensional Euclidean space $\mathbb{R}^d$. We describe a technique singling out the most promising directions which pivots should lie on. To the best of our knowledge this is the first work considering the pivot placement problem.

The remainder of the paper is organized as follows. Section 2 describes the pivot placement technique. Section 3 compares the approach here introduced with competitors. Finally, Section 4 draws the conclusions.

Algorithm 1: PPP(D, k)

Input: D, set of objects; k, number of required pivots

$n = |D|$
// Compute small clusters
$K = K_0$ *// number of small clusters*
$C = GetSmallClusters(D, K)$
// Compute intra-cluster directions
$N = \frac{K(K+1)}{2}$ *// number of directions*
for $i = 1$ *to* K **do**
  $\mathbf{v}_i = \mathrm{PCA}(C_i)$
  $w_i = \frac{|C_i|(|C_i|-1)}{n(n-1)}$
  $\mathbf{c}_i = \frac{1}{|C_i|}\sum_{x \in C_i} x$
// Compute inter-cluster directions
$l = K + 1$
for $i = 1$ *to* $K - 1$ **do**
  for $j = i + 1$ *to* K **do**
    $w_l = \frac{|C_i|(|C_j|-1)}{n(n-1)/2}$
    $\mathbf{v}_l = \frac{\mathbf{c}_i - \mathbf{c}_j}{\|\mathbf{c}_i - \mathbf{c}_j\|}$
    $l = l + 1$
// Compute angles between directions
for $i = 1$ *to* N*-1* **do**
  for $j = i$ *to* N **do**
    $\alpha_{i,j} = \arccos(|\langle \mathbf{v}_i, \mathbf{v}_j \rangle|)$
    $\alpha_{j,i} = \alpha_{i,j}$
// Perform prioritized fixed-width clustering
$\theta = \theta_0$ *// cone angle*
$\langle \ell, c \rangle = \mathrm{PFWC}(\boldsymbol{\alpha}, w, \theta)$
// Determine the best pivots
$M = \frac{1}{n}\sum_{i=1}^{n} x_i$
$L = \max_{i=1}^{n} \|x_i - M\|$
$T = T_0$ *// pivot displacement*
for $i = 1$ *to* $\min\{c, k\}$ **do**
  $J_i = \{j : \ell_j = i\}$
  $p_i = M - T \cdot L \cdot \frac{\sum_{j \in J_i} w_j \mathbf{v}_j}{\sum_{j \in J_i} w_j}$
return p

Algorithm 2: PFWC($\boldsymbol{\alpha}$, w, θ)

Input: $\boldsymbol{\alpha}$, pairwise angles; w, direction weights; θ, angle threshold
Output: ℓ, labels (cluster numbers) assigned to directions; c, number of obtained clusters

for $i = 1$ *to* N **do**
  for $j = 1$ *to* N **do**
    $\beta_{i,j} = 0$
    if $\alpha_{i,j} \leq \frac{\theta}{2}$ **then**
      $\beta_{i,j} = 1$
$m = N$
$c = 0$
while $m > 0$ **do**
  $c = c + 1$
  $\varrho = \boldsymbol{\beta} \cdot w$
  $i = \arg\max_i \varrho_i$
  for $j = 1$ *to* N **do**
    if $\beta_{i,j} = 1$ **then**
      $m = m - 1$
      $\ell_j = c$
      $\beta_{j,\cdot} = 0$
      $\beta_{\cdot,j} = 0$
return $\langle \ell, c \rangle$

2 Principal Directions-Based Pivot Placement Algorithm

In this section, the PPP (for Principal directions-based Pivot Placement) algorithm is presented. The pseudo-code of the algorithm is reported in figure. It receives in input the dataset D and the number k of required pivots.

First, data is partitioned into a number K of homogeneous clusters $C_1, \ldots, C_K$, also called *small clusters* in the sequel. With this aim, the K-means algorithm is employed, that outputs a controlled number of prototypes having the property of minimizing the average distance to the associated groups. The small clusters are then exploited to determine directions connecting the data objects. A set $\mathbf{v}_1, \ldots, \mathbf{v}_N$ of $N = \frac{K(K+1)}{2}$ *directions*, that are versors associated with either a single small cluster ($1 \leq i \leq K$) or a pair of small clusters ($K < i \leq N$), is

populated. Each direction has also a *weight* w_i which represents the significance of the direction.

Directions $\mathbf{v}_i$ associated with single small clusters intend to capture the main direction along which the cluster objects spread. This direction is naturally captured by the *first principal component* of the cluster which is computed by exploiting Principal Component Analysis (PCA). PCA is [9] an orthogonal linear transformation to a new coordinate system such that the greatest variance by any projection of the data comes to lie on the first coordinate (called the first principal component). The weight w_i of the direction $\mathbf{v}_i$ consists in the number of pairwise objects of the cluster. As for the directions $\mathbf{v}_l$ ($K < l \leq N$) associated with pairs C_i and C_j of different small clusters, they are defined in the terms of the vector linking the center of mass $\mathbf{c}_i$ of C_i and $\mathbf{c}_j$ of C_j, and the associated weight is given by the number of distinct pairs consisting of objects coming from the two clusters. Once intra-cluster and inter-cluster directions have been obtained, together with their importance, the matrix $\boldsymbol{\alpha}$ of pairwise angles $\alpha_{i,j}$ ($1 \leq i, j \leq N$) formed by the directions $\mathbf{v}_i$ and $\mathbf{v}_j$ is computed, where $\alpha_{i,j} = \arccos(|\langle \mathbf{v}_i, \mathbf{v}_j \rangle|)$, in order to be exploited to determine directions along which pivots will be placed.

With this aim directions are grouped according to a *prioritized fixed width clustering strategy* (see algorithm PFWC in figure). Specifically, at each main iteration of PFWC, a seed direction $\mathbf{v}_i$ is selected and all the directions $\mathbf{v}_j$ (not already assigned to a group) such that $\alpha_{i,j} \leq \theta/2$ are assigned to the same group. These are the directions lying in the double cone having apex in the origin and axis collinear to $\mathbf{v}_i$ and forming an angle between surface and axis of $\theta/2$ radians. Seed directions are selected by assigning a *priority* ϱ_i ($1 \leq i \leq N$) to those not already grouped and, then, by selecting the direction scoring the maximum priority. Priorities are computed as $\varrho = \boldsymbol{\beta} \cdot w$, that is to say ϱ_i is the sum of the weights associated with the directions that will become part of the group obtained by using direction $\mathbf{v}_i$ as seed.

Let M and L be the center of mass and the radius, respectively, of the dataset D. Moreover, let T a positive number, also called *displacement*, which is used to locate the hyper-sphere $\mathcal{S}$ of radius $T \cdot L$ centered in M. The c groups of directions returned by PFWC are finally exploited in order to place pivots. Pivot p_i ($1 \leq i \leq k$) is obtained from the i-th group of directions, that are the directions $\mathbf{v}_j$ such that $\ell_j = i$. Specifically, p_i corresponds to one of the two points located at the intersection of the surface of the sphere $\mathcal{S}$ and the straight line passing through M whose direction is given by the mean of the versors in the i-th group.

3 Experiments

We used some collections of data available in the *Metric Spaces Library* [8] and in the *UCI KDD Repository* [3]: *COLOR* (112,682 points with 112 features), *LANDSAT* (275,465 points with 60 features), and *NASA* (40,150 points with 20 features). The PPP method has been compared with the *Selection of Random*

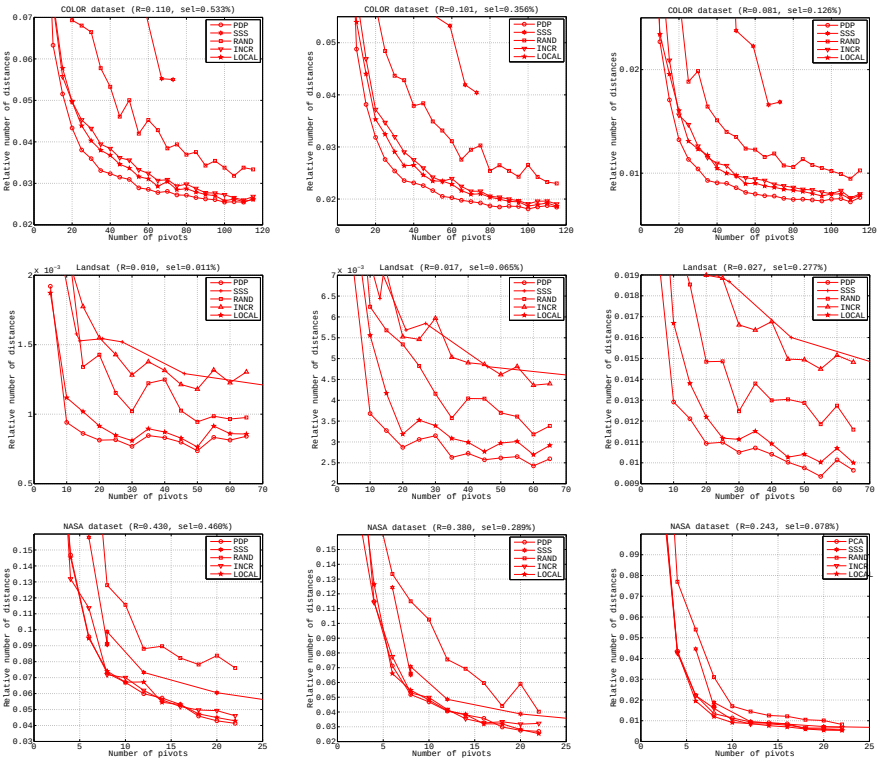

Fig. 1: Experimental results.

Groups (RAND), *Incremental Selection* (INCR), *Local Optimum Selection* (LOCAL), and *Sparse Spatial Selection* (SSS) strategies. The parameters of PPP employed are the following: number of small clusters $K_0 = 100$, cone angle $\theta_0 = \pi/3$, and pivot displacement $T_0 = 10$. As for the parameters of the competitors, we set them to the values suggested in [4]. Specifically, for RAND the number of sets of pivots is $N = 50$; for INCR the size of the sample is $N = 50$; for LOCAL the number of pairs A is set to 100,000, the number of iterations N' is set to k and the size of the sample is $X = N - 1$; finally, for SSS α has been set so that the number of selected pivots were close to k.

We performed a number of range query searches. Specifically, for each dataset, we considered three different radiuses. Radius values are such that on average the 0.01%, 0.05% and 0.1% of the dataset objects are selected. As for the queries, 5,000 objects have been picked out at random from the dataset. Experimental results are reported in Figure 1. We varied the number of pivots up to about the number of dimensions (reported on the x-axis) and compared the number of distances computed during both the *candidate selection* and the *filtering phase*, reported on the y-axis. Figures highlight that PPP performs better than com-

petitors on the COLOR and LANDSAT datasets, and comparably to the other methods (INCR and LOCAL) on the NASA dataset. Experiments are encouraging, since they confirm that PPP may exhibit state-of-the-art performances as a method for pivot selection.

4 Conclusions

We addressed the pivot placement problem and provided a suitable algorithm to deal with it. Experiments confirmed that the proposed method, based on the idea of placing the pivots in the space by determining directions which achieve the better alignment with the whole dataset, improves indexing effectiveness. As for the future work, we are going to preform a more extensive experimental campaign, including further datasets and a comprehensive parameter sensitivity analysis. Also, theoretical assessment of the proposed strategy and extending it to general metric spaces is of interest.

References

1. F. Angiulli and F. Fassetti. Indexing uncertain data in general metric spaces. *IEEE Trans. Knowl. Data Eng.*, 24(9):1640–1657, 2012.
2. L. Ares, N. Brisaboa, M. Esteller, O. Pedreira, and A. Places. Optimal pivots to minimize the index size for metric access methods. In *International Workshop on Similarity Search and Applications (SISAP)*, pages 74–80, 2009.
3. K. Bache and M. Lichman. UCI machine learning repository, 2013.
4. B. Bustos, G. Navarro, and E. Chávez. Pivot selection techniques for proximity searching in metric spaces. *Pattern Recognition Letters*, 24(14):2357–2366, 2003.
5. B. Bustos, O. Pedreira, and N. R. Brisaboa. A dynamic pivot selection technique for similarity search. In *ICDE Workshops*, pages 394–401, 2008.
6. E. Chávez, J. L. Marroquín, and R. A. Baeza-Yates. Spaghettis: An array based algorithm for similarity queries in metric spaces. In *Symp. on String Processing and Information Retrieval (SPIRE)*, pages 38–46, 1999.
7. E. Chávez, G. Navarro, R. Baeza-Yates, and J. Marroquín. Searching in metric spaces. *ACM Computing Surveys*, 33(3):273–321, 2001.
8. K. Figueroa, G. Navarro, and E. Chávez. Metric spaces library, 2007. Available at `http://www.sisap.org/Metric_Space_Library.html`.
9. I. T. Jolliffe. *Principal Component Analysis*. Springer, 2^{nd} edition, October 2002.
10. L. Micó, J. Oncina, and E. Vidal. A new version of the nearest-neighbour approximating and eliminating search algorithm (aesa) with linear preprocessing time and memory requirements. *Pattern Recognition Letters*, 15(1):9–17, 1994.
11. O. Pedreira and N. R. Brisaboa. Spatial selection of sparse pivots for similarity search in metric spaces. In *Proceedings of the 33rd conference on Current Trends in Theory and Practice of Computer Science*, pages 434–445, 2007.
12. H. Samet. *Foundations of Multidimensional and Metric Data Structures*. Morgan Kaufmann Publishers Inc., 2005.
13. P. N. Yianilos. Data structures and algorithms for nearest neighbor search in general metric spaces. In *ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 311–321, 1993.
14. P. Zezula, G. Amato, V. Dohnal, and M. Batko. *Similarity Search: The Metric Space Approach*, volume 32 of *Advances in Database Systems*. Springer, 2006.