A Spline-based Cost Model for Metric Trees

Marcos V. N. Bedo[1,2], **Agma J. M. Traina**[2] **and Caetano Traina Jr.**[2]

[1]Fluminense Northwest Institute – Fluminense Federal University (UFF)
St. A. Pádua – RJ – Brazil

[2]Institute of Mathematics and Computer Science – University of São Paulo (USP)
São Carlos – SP – Brazil

{bedo,agma,caetano}@icmc.usp.br

Abstract. *Whenever two (or more) access methods are alternatives for the execution of a query, how to choose which one is the best for the task? Such a decision is made by the DBMS optimizer module, which models the query costs according to the distribution of the data space. Cost modeling of similarity searches, however, requires the representation of distances' rather than data distribution. In this paper, we propose the Stockpile model for cost estimation of similarity queries on metric trees by using pivot-based distance histograms that represent the local densities around the query elements. By combining the local densities to the probability of traversing the tree nodes, Stockpile provides a fair estimation of both disk accesses (I/O costs) and distance calculations (CPU costs). We compared Stockpile and two literature models regarding similarity queries in real-world data sources and our model was up to* 85% *more precise than the competitors.*

1. Introduction

Similarity searching is a foundational paradigm for many computer applications, such as content-based retrieval, classification, clustering, and data visualization [Zezula et al. 2006]. In practice, two of the most requested similarity operations are the range and k-NN searches. An example of range query is **(Q1)** "List the bottles in the wine cellar whose combination of fixed and volatile acidities differs at most 4 mg/L to this Italian wine", while a k-NN query example is **(Q2)** "Find the 3 closest cabs to this restaurant". Range and k-NN queries can be modeled upon a *metric space*, where the elements (bottles and cabs, in the examples) are represented as points and the (dis)similarity between each pair of points is evaluated by a distance function.

Formally, a metric space is a pair $\mathcal{M} = \langle \mathbb{S}, \delta \rangle$, where $\mathbb{S}$ is the domain of the points and δ is a metric that complies with the properties of symmetry, non-negativity, and triangular inequality. Accordingly, given a data source $S \subseteq \mathbb{S}$, a query element $s_q \in S$ and a threshold $\xi \in \mathbb{R}_+$, a range query Rq retrieves every element in S within the closed ball centered at s_q with radius ξ such that $Rq(S, s_q, \xi) = \{s_i \in S \mid \delta(s_i, s_q) \leq \xi\}$. On the other hand, a k-NN query returns a quantity $k \in \mathbb{N}$ of elements whose distance to the query element s_q are the smallest. In an equivalent way, a k-NN query can be seen as a variation of the range query, i.e., a range search with a set radius ξ such that $|Rq| = k$ [Tasan and Özsoyoglu 2004].

Several indexing schemes in the form of metric access methods have been proposed to speed up similarity searching [Lokoč 2010]. Particularly, *tree-based* methods stand out as the most suitable strategies for the indexing of very large data sources as they organize the search space into a hierarchical and balanced fashion. Remarkable tree-based structures include the M-Tree [Ciaccia et al. 1997] and its variations, such as Slim-Tree [Traina Jr. et al. 2002] and PM-Tree [Skopal et al. 2004]. Although tree-based indexes focus on minimizing both distance calculations (through the clustering of the elements) and disk accesses (by using database paging principles), their performances differ depending on the query. Therefore, given a similarity query over a data source indexed by two or more access methods, a database query optimizer must decide which method will be employed to execute the search. Such a query optimizer' decision is made upon a *cost model* for metric trees.

Thus far, most of the research effort has addressed the cost of similarity queries regarding multidimensional spaces [Korn et al. 2001, Tao et al. 2004]. However, besides inapplicable to non-dimensional metric spaces, many of these models are also unable to handle cost estimation of k-NN queries. One of the challenging aspects of modeling k-NN searches is no limiting radius is known beforehand. Accordingly, existing k-NN models estimate a threshold as the query radius by using the pairwise distance distribution within S. Such estimates follow a *biased assumption*: query elements are more likely posed in high-density areas of the search space. For instance, the model for multidimensional spaces in [Aly et al. 2015] assumes the k-NN radii follow a uniform distribution regarding fixed intervals of k, while the models in [Ciaccia et al. 1998] and [Baioco et al. 2007] assume k-NN radii follow a binomial and an exponential distribution, respectively. The main drawback of such models is they disregard the 'locality' of each query, i.e. they rely on a global and pairwise distance distribution without properly considering the density around each query element.

In this study, we propose the *Stockpile cost model* for the estimation of similarity searching costs in tree-based methods. Stockpile distinguishes itself from previous models as it estimates the cost of range queries according to their locality. The overall idea of our approach is building a small set of pivot-based distance histograms in such a way the densities around query elements are represented. Additionally, by using pivot-based histograms to predict the k-NN radii, the local densities also are taken into account in the cost estimation of k-NN searches. Stockpile histograms are built as splines on pivot-based distance distributions according to a fixed number of buckets so that they easily fit into main memory. Aiming at evaluating the accuracy of Stockpile, we compared our approach to models in [Ciaccia et al. 1998] and [Baioco et al. 2007] regarding real-world data sources and results showed Stockpile outperforms both competitors in terms of accuracy. Accordingly, the main contributions of the paper are as follows:

- We introduce the Stockpile cost model, which estimates the cost of similarity queries by using the locality of the query element,
- We experimented with our approach over real-world data sources and results indicate Stockpile predictions are more accurate than those obtained by models in [Ciaccia et al. 1998] and [Baioco et al. 2007].

The remainder of the paper is organized as follows. Section 2 summarizes related work. Section 3 introduces Stockpile and its parameters. Sections 4, 5 and 6 show the results of evaluations performed, while Section 7 concludes the paper.

2. Background and Related Work

2.1. M-Tree and Slim-Tree

The M-Tree [Ciaccia et al. 1997] is a dynamic structure that hierarchically organizes the elements within a metric space into closed balls, which are stored as *nodes*. Basically, M-Tree uses two types of nodes ($\mathtt{N}$), namely directory ($\mathtt{N}_d$) and leaf ($\mathtt{N}_l$) nodes. Directory nodes store a set of balls, while leaf nodes store the indexed elements themselves. Accordingly, a leaf node $\mathtt{N}_l$ has the format $\mathtt{N}_l(s_j) = \{\langle s_i, d_{ij}\rangle\}$, where s_i is a data element such that $s_i \in S \subseteq \mathbb{S}$, and d_{ij} is the distance of s_i to the rooting element s_j (parent) of the leaf node. A directory node $\mathtt{N}_d$ has the format $\mathtt{N}_d(s_j) = \{\langle s_i, \xi_i, \mathtt{N}(s_i), d_{ij}\rangle\}$, where $s_i \in S$ is the rooting element to the subtree $\mathtt{N}(s_i)$, ξ_i is the covering radius of $\mathtt{N}(s_i)$, and d_{ij} is the distance of s_i to rooting element s_j. Precomputed distances d_{ij} are employed for the pruning of nodes/elements when a similarity query is executed [Traina Jr. et al. 2002, Skopal et al. 2004]. The top directory node is the root node: it has no parent and covers all indexed elements. Figure 1 shows two M-tree examples under the L_2 distance.

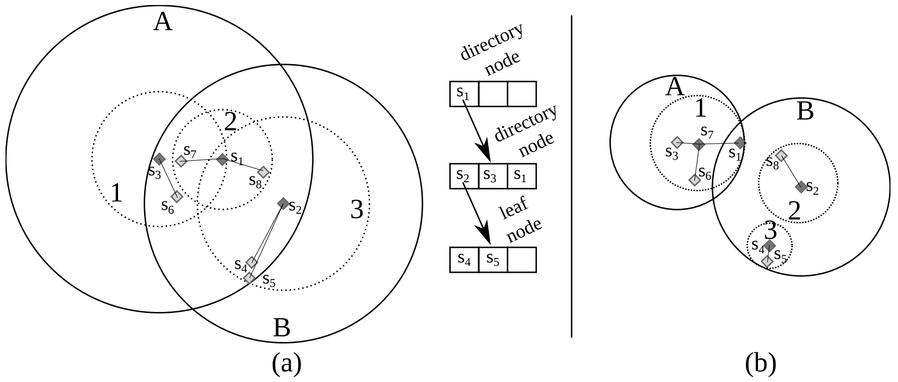

Figure 1. Two M-Trees indexing the same elements. (a) M-Tree with a large overlap, and (b) M-Tree with a smaller overlap.

Distinct partitioning strategies can be applied for the construction of a valid M-Tree such that the nodes may overlap, but all elements rooted by s_j are within the covering radius of $\mathtt{N}(s_j)$. Notice, however, the larger the "volume"[1] of the tree, the higher the probability of occurring intersections among the nodes. For instance, although both partitions of Figures 1(a) and (b) generate valid M-Trees, the tree in Figure 1(b) has a smaller volume in comparison to the tree in Figure 1(a). The Slim-Tree method extends the M-Tree by using improved partitioning algorithms that minimize the volume of the tree through the evaluation of the overlaps in terms of a single measure, called *fat-factor* [Traina Jr. et al. 2002]. Formally, the fat-factor of a given tree T regarding a data source S is defined by Equation 1.

$$fat(T) = \frac{I_C - |S| \cdot h}{|S|} \cdot \frac{1}{m - h} \tag{1}$$

where I_c denotes the sum of node accesses for the execution of point queries, i.e. $Rq(S, s_i, 0.0)$, for every $s_i \in S$, h is the height of the tree, and m is the total number

[1]We quoted "volume" as there is no universal notion of volume in metric spaces. In practice, the number of elements covered by the ball can be used as a suggestion for the volume.

of nodes. The fat-factor expresses how "good" is the tree in a $[0, 1]$ scale with regards to the overlapping of elements at the same level. For instance, the fat-factor of the tree in Figure 1(a) is 0.5, while the same measure for Figure 1(b) is ≈ 0.04. In this scenario, the tree in Figure 1(b) is more likely to avoid duplicate scans of the search space in comparison to the tree in Figure 1(a) when executing a query.

2.2. Costs models for range and k-NN queries

The relationship between elements is expressed by means of distances within metric trees. Therefore, estimating a query cost involves the representation of the *distance distribution* on S regarding a metric δ. Basically, two types of distance distributions can be gathered from a metric tree, namely the pairwise and the pivot-based distance distribution. A *pairwise distance distribution* $\mathcal{T}$ is a *single* distribution that captures the frequency of distances between every pair of elements in S. Therefore, a joint and normalized pairwise distance distribution $\mathcal{T}^{\mathcal{C}}$ can be seen as a probability function $F(x)$ so that $F(x) = \texttt{Prob}\{\delta(s_i, s_j) \leq x\},\ \forall\ s_i, s_j \in S$. The proposal in [Ciaccia et al. 1998] follows this rationale and assumes the cost of similarity queries can be estimated by a *biased* query model, which implies that the distances between the query element and points in S are supposed to follow $\mathcal{T}^{\mathcal{C}}$. In this case, given a range query $Rq(S, s_q, \xi)$, the probability of scanning a node rooted by s_i with radius ξ_i of a tree-based method T that indexes S is expressed as Equation 2.

$$\texttt{Prob}\{\delta(s_q, s_i) \leq \xi + \xi_i\} \approx F(\xi + \xi_i) \tag{2}$$

The number of scanned tree nodes is estimated by the sum of the probability of accessing each node as in Equation 3. Analogously, the number of comparisons between s_q and the element in T is the sum of the weighted probabilities of accessing each node, where the weight is the number of entries in that node as in Equation 4.

$$\texttt{nodes\_scanned}_{Ciaccia}(T, s_q, \xi) \approx \sum_{i}^{m} F(\xi + \xi_i) \tag{3}$$

$$\texttt{distances\_calculated}_{Ciaccia}(T, s_q, \xi) \approx \sum_{i}^{m} |\texttt{N}(s_i)| \cdot F(\xi + \xi_i) \tag{4}$$

The authors in [Ciaccia et al. 1998] also propose the use of $F(x)$ as part of a binomial probability function so that an estimated radius ξ for a k-NN query can be drawn from $F(x)$. However, their approach has two major drawbacks, namely *(i)* the pairwise distance distribution is expensive to obtain, and *(ii)* the cost of the estimations depends on the number of nodes.

Aiming at avoiding such drawbacks, the study in [Baioco et al. 2007] proposes a model that generalizes the use of the *fractal dimension* for the estimation of query costs. The proposal extends the previous approach of [Korn et al. 2001] for metric trees, where the joint pairwise distance distribution can be constructed following the box-counting algorithm. The authors argue the fractal dimension is a good approximation of the intrinsic dimension and can be calculated through the *Distance-Plot* [Korn et al. 2001, Navarro et al. 2017]. Such a graphic plots $\mathcal{T}^{\mathcal{C}}$ by using $log \times log$ axes and enables the approximation of the joint frequencies by a linear function. Thus, the model assumes whenever $\mathcal{T}^{\mathcal{C}}$ is a joint exponential distribution, then the slope of the fitting line approximates the fractal dimension $\mathcal{D}$ of S. Under such conditions, the number of scanned regions and distance calculations of a range query $Rq(S, s_q, \xi)$ are given by Equations 5 and 6, respectively.

$$\texttt{nodes\_scanned}_{Baioco}(T, s_q, \xi) \approx \frac{1}{R^{\mathcal{D}}} \sum_{i}^{h} |S|^{\frac{i}{h}} \left(\sqrt[\mathcal{D}]{|S|^{\frac{-i}{h}}} + \xi \right)^{\mathcal{D}} \tag{5}$$

$$\texttt{distances\_calculated}_{Baioco}(T, s_q, \xi) \approx \frac{1}{R^{\mathcal{D}}} \sum_{i}^{h} |S|^{\frac{i+1}{h}} \left(\sqrt[\mathcal{D}]{|S|^{\frac{-i}{h}}} + \xi \right)^{\mathcal{D}} \tag{6}$$

where R is the covering radius of the entire tree. The authors also claim the distance between the query element and its k^{th} neighbor can be calculated by using $\mathcal{D}$, which enables the transformation of k-NN into range queries. The model in [Baioco et al. 2007] considers k-NN overestimated radii are preferable to underestimated ones, a subject deeply discussed in [Vieira et al. 2007]. Therefore, the model adds a controlled overestimation based on the characteristics of the tree, i.e. the fat-factor of the index, and scales Equations 5 and 6 by the constant $(1 + fat(T))$.

Other models also rely on a single and biased representation of the pairwise distance distribution. For instance, the study in [Tao et al. 2004] extends the binomial approach in [Ciaccia et al. 1998] for k-NN cost estimation in low dimensional spaces, while the proposal in [Lu et al. 2014] estimates the query costs by combining S to other domains. Recently, the study in [Aly et al. 2015] introduced the *Staircase* model for k-NN queries in multidimensional spaces. The authors argue the cost of such queries is *stable*, i.e., the cost of executing a k-NN query with larger k can be the same of executing a query with a smaller k as the same nodes of the index are accessed and provided the incremental k-NN searching procedure is employed [Hjaltason and Samet 2003]. The model assumes a uniform distribution for fixed intervals of k, which results in a compact representation of the indexed data. However, distance distributions hardly follow the uniform assumption [Korn et al. 2001], which harm the usability of the model in tree-based methods.

Notice the biases of the reviewed models are mainly related to the adoption of a single representation of the pairwise distance distribution. The study in [Tasan and Özsoyoglu 2004] comes up with a suggestion to avoiding such a bias in the task of predicting k-NN radii. Basically, the authors propose the gathering of distribution $\mathcal{T}$ in the form of a histogram $H(x)$, $H(x) \approx \mathcal{T}$ so that no inference on the type of the distance distribution itself is required. Therefore, the radius ξ' of a k-NN query can be straightforwardly estimated by Equation 7.

$$\int_{0}^{\xi'} H(x)d(x) = k \tag{7}$$

Moreover, the authors show the *unique* representation of the pairwise distribution can be replaced by pivot-based distance distributions. Such an observation is similar to the “Homogeneity of Viewpoints” property in [Ciaccia et al. 1998], but in

[Tasan and Özsoyoglu 2004] the authors suggest keeping the pivot-based distributions instead of replacing them by the pairwise distribution. Formally, a pivot-based distribution $\mathcal{T}_p$ on S for a pivot $p \in \mathcal{P} \subseteq S$ follows the Definition 2.1.

Definition 2.1 (Pivot-based distance distribution – $\mathcal{T}_p$) *Given a data source S, a metric δ, and a pivot $p \in \mathcal{P}$, $\mathcal{T}_p$ captures the distance from each $s_i \in S$ to p. Distance value set $\mathcal{V}_p$ contains the distinct and sorted values of $\delta(s_i,\ p)$, i.e., $\mathcal{V}_p = \{v_p(j)\ :\ 1 \leq j < m_p,\ m_p \leq |S|\}$, where $v_p(m_p)$ is the largest distance between any s_i to p. Frequency $f_p(j)$ is the number of elements of S whose distance $\delta(s_i,\ p) = v_p(j)$. The pairs $\{\langle v_p(j), f_p(j)\rangle,\ \forall\ v_p(j) \in \mathcal{V}_p\}$ constitute $\mathcal{T}_p$, i.e. $\mathcal{T}_p = \{\langle v_p(1), f_p(1)\rangle, \ldots, \langle v_p(m_p), f_p(m_p)\rangle\}$. $\mathcal{T}_p^+$ is the extension of $\mathcal{T}_p$ to the entire domain of distances by setting 0 as the frequency for any $v_p \in \mathbb{R}_+ \setminus \mathcal{V}_p$.*

Equally spaced Equi-Width histograms can be employed for the approximation of $\mathcal{T}_p$ such that $H_p(x) \approx \mathcal{T}_p$. More sophisticated histograms enable the bounding of the approximation error as in V-Optimal [Ioannidis 2003], Curve-Fitting [König and Weikum 2002], or Compact-Distance Histograms [Bedo et al. 2015]. Particularly, a Compact-Distance Histogram (CDH) can be seen as a continuous piecewise linear function whose squared error to $\mathcal{T}_p$ is minimal so that optimal CDHs are obtained by using the squared error as the optimization target in the Bellman-Ford algorithm [Ioannidis 2003]. Therefore, given a k-NN query, a set of pivots $\mathcal{P} = S$, and a set of pivot-based distance histograms $H_p(x)$, Equation 7 can be rewritten as Equation 8.

$$k = \texttt{Prob}(p) \cdot \int_0^{\xi'} H_p(x)dx \tag{8}$$

where `Prob(p)` is a binary probability of p having the same distance distribution of the query element of the k-NN search. Although theoretically removing many of the biases from previous models, the estimations in [Tasan and Özsoyoglu 2004] still have two practical setbacks. First, both pairwise distance distribution and the set of pivot-based distributions are needed, which entails an expensive construction cost. Last but not least, it is not always possible to set $\mathcal{P} = S$ due to memory constraints.

In this paper, we follow the indications of both [Ciaccia et al. 1998] and [Tasan and Özsoyoglu 2004] for the creation of a cost model that takes into account the locality of query elements and that comply with memory constraints.

3. The Stockpile cost model

In this section, we propose a cost model for predicting the number of disk accesses and distance calculations for any range or k-NN query to be executed by a metric tree. The model itself relies on two previously computed data structures, namely *(i)* a set of histograms and *(ii)* meta-statistics, e.g. the fat-factor, about the metric tree. Such structures are typically kept in main memory as a *pile* of resources (and hence the name *Stockpile* for the model) to be evaluated on-the-fly according to user-posed queries, which requires the model structures to be parameterized in terms of space constraints. Histograms are especially suitable for this scenario, as they enable the use of the constraint *maximum number of buckets* ($\mathcal{B}$) for the representation of the distributions. Stockpile employs Compact-Distance Histograms (CDHs) for the approximation of pivot-based distance distribution as splines. Accordingly, we denote a CDH by $f_p(x)$ such that $f_p(x) \approx \mathcal{T}_p$. Figure 2(a) shows an example of a $\mathcal{T}_p$, while Figure 2(b) presents three possible CDHs by using $\mathcal{B} = 4, 5$ and 6 buckets, respectively.

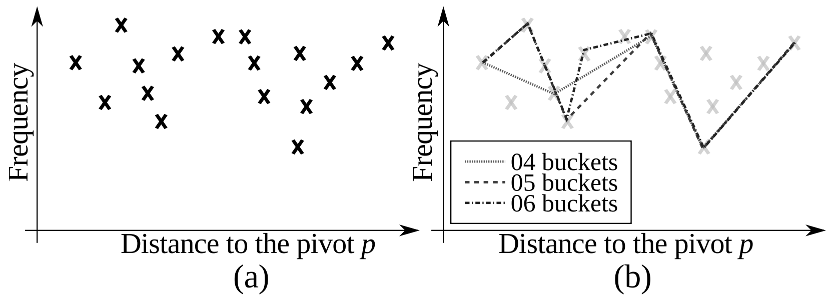

Figure 2. Summarization of a distance distribution $\mathcal{T}_p$ regarding a piecewise linear function $f_p(x)$. (a) Original pivot-based distance distribution, and (b) Compact-Distance Histogram for a maximum of $4, 5$ and 6 buckets.

Notice the *number of pivots* must be limited according to available memory so that constraints $|\mathcal{P}|$ and $\mathcal{B}$ are balanced somehow. Although an exhaustive search of trade-offs can be used for finding such a balance, we argue that a representative number of pivots is given by the intrinsic dimension, which is the rationale employed in several pivot-based problems [Bustos et al. 2003]. Thus, our model uses $|\mathcal{P}| = \mathcal{D}$ and obtains the value of the constraint $\mathcal{B}$ according to the remaining memory space.

The major strength of Stockpile is it takes into account the "locality" of the query element by assuming *pivots closer to query points are more likely to resemble the distance distributions of the query elements.* The premise is especially fair whenever the density of distances around the pivot is uniformly distributed according to $f_p(x)$. Therefore, we assume the probability of the query element having the same distribution of a given pivot is linear with regards to the distance between them. Formally, let s_q be a query element and $\mathcal{P}$ be a set of pivots, the probability $\texttt{Prob}(p)$ of s_q resembling the distribution $f_p(x)$ is proportional to $\delta(s_q, p)$ so that $\texttt{Prob}(p) = d(s_q, p)/C_2$, where $d(s_q, p) = C_1 - \delta(s_q, p)$. Both C_1 and C_2 are local constants that depend on the query element s_q. Such constants are calculated as $C_1 = \sum_{p \in \mathcal{P}} \delta(s_q, p)$ and $C_2 = \sum_{p \in \mathcal{P}} d(s_q, p)$. The joint probability of the query element resembling the pivots in $\mathcal{P}$ is given by Equation 9.

$$\sum_{p \in \mathcal{P}} \texttt{Prob}(p) = \frac{d(s_q, p_1)}{C_2} + \frac{d(s_q, p_2)}{C_2} + \cdots + \frac{d(s_q, p_{|\mathcal{P}|})}{C_2} = \frac{C_1(|\mathcal{P}| - 1)}{C_1(|\mathcal{P}| - 1)} = 1 \tag{9}$$

Accordingly, Stockpile models the probability of traversing a node as the linear combination of the local information and the densities given by the CDHs.

3.1. Costs estimation of range queries

Suppose a range query $Rq(S, s_q, \xi)$ to be executed by a metric tree T. All leaf nodes in T that intercept the query ball defined by $\langle s_q, \xi \rangle$ must be evaluated because their elements are potentially inside the query ball. Root nodes to these leaf nodes must be evaluated as well. Therefore, the local probability of accessing a node $\texttt{N}(s_j)$ regarding a given pivot p is modeled upon the covering radius of the node (ξ_j) and the range query radius (ξ) as expressed by Equation 10.

$$\begin{aligned}\texttt{Prob}(\text{node is accessed}) &= \texttt{Prob}\{\delta(s_q, p) \leq \xi_j + \xi\} \\ &\approx \bar{F}_p(\xi_j + \xi) = \frac{\displaystyle\int_0^{\xi_j+\xi} f_p(x)dx}{\displaystyle\int_0^{v_p(m_p)} f_p(x)dx}\end{aligned} \tag{10}$$

Naturally, the local probability is 1 whenever $\xi_j + \xi > v_p(m_p)$. The overall probability of accessing a node of T is given by each pivot $p \in \mathcal{P}$ and the joining of Equations 9 and 10 into Equation 11. Similarly, Stockpile combines Equations 9 and 11 into Equation 12 for the estimation of distance calculations in range queries.

$$\texttt{nodes\_scanned}(T, s_q, \xi) \approx \sum_j^m \sum_{p \in \mathcal{P}} \texttt{Prob}(p) \cdot \bar{F}_p(\xi + \xi_j) \tag{11}$$

$$\texttt{distances\_calculated}(T, s_q, \xi) \approx \sum_j^m |\texttt{N}(s_j)| \cdot \left(\sum_{p \in \mathcal{P}} \texttt{Prob}(p) \cdot \bar{F}_p(\xi + \xi_j) \right) \tag{12}$$

The intuition in Equation 12 is the number of distance calculations is proportional to the probability of accessing each node, where $|\texttt{N}(s_j)|$ is either the number of entries (in the case of directory nodes) or the number of elements (in the case of leaf nodes).

3.2. Cost estimation of *k*-NN queries

The cost estimation of k-NN queries requires an additional step in comparison to range queries. Such an extra step is the "reduction" of the k-NN query to a range query by setting the query threshold ξ as the distance between the query element and its k^{th} neighbor. However, such a distance is unknown until the effective execution of the k-NN query and, consequently, Stockpile must estimate the query radius before calculating the query costs. Formally, given a k-$NN(S, s_q, k)$ query and a CDH related to pivot p, Stockpile estimates the distance between s_q and its k^{th} neighbor as the threshold ξ'_p according to Equation 13.

$$\frac{|S|}{\int_0^{v_p(m_p)} f_p(x)dx} \cdot \int_0^{\xi'_p} f_p(x)dx = k \tag{13}$$

where the term $(|S|)/(\int_0^{v_p(m_p)} f_p(x)dx)$ is the uniform distribution of the histogram approximation error among its buckets. Notice the solving of Equation 13 can be carried out by a numeric method, such as Newton-Raphson, as the primitive function for $f_p(x)$ is continuous and monotonically crescent. Accordingly, Stockpile combines the probability of selecting the pivot in Equation 9 to Equation 13 so that k-$NN(S, s_q, k)$ is reduced to a range query whose radius depends on $\mathcal{P}$. The number of T scanned nodes regarding a k-NN query is given by Equation 14, whereas the number of distance calculations is estimated as in Equation 15.

$$\texttt{nodes\_scanned}(T, s_q, k) \approx \sum_j^m \sum_{p \in \mathcal{P}} \texttt{Prob}(p) \cdot \left(\frac{k}{|S|} + \frac{\displaystyle\int_{\xi'_p}^{\xi'_p+\xi_j} f_p(x)dx}{\displaystyle\int_0^{v_p(m_p)} f_p(x)dx} \right) \tag{14}$$

$$\texttt{dist\_calc}(T, s_q, k) \approx \sum_{j}^{m} |\texttt{N}(s_j)| \cdot \left[\sum_{p \in \mathcal{P}} \texttt{Prob}(p) \cdot \left(\frac{k}{|S|} + \frac{\displaystyle\int_{\xi'_p}^{\xi'_p + \xi_j} f_p(x)dx}{\displaystyle\int_{0}^{v_p(m_p)} f_p(x)dx} \right) \right] \tag{15}$$

3.3. Optimistic *vs.* pessimistic estimates from Stockpile

The proposed estimates are particularly accurate for *optimal metric trees*, i.e. trees with such a partitioning that generate nodes with no overlap. Accordingly, we call *optimistic estimates* the Stockpile predictions of Equations 11, 12, 14 and 15. However, indexing of real data sources typically results in a non-negligible number of overlaps between the tree nodes. Therefore, a slight overestimation in the number of disk accesses and distance calculations may provide a better approximation for the query real costs [Vieira et al. 2007]. In this scenario, Stockpile uses the fat-factor of T for the calculation of the cost overestimation, which results in the scaling of Equations 11, 12, 14 and 15 by the $(1 + fat(T))$ constant. We call these Stockpile scaled predictions *pessimistic estimates*. Optimistic/pessimistic estimates can be easily set for each metric tree T in a query optimization environment of a database search engine.

4. Experiments

This section reports on a set of experiments performed over four real-world data sources with low to medium dimensionality, as detailed in Table 1. Data sources `CITIES`[2], `WINE`[3], `LETTER`[3], and `CANVAS`[4] were queried by using well-known metrics of the Minkowski family (L_p) and indexed by Slim-Trees. Stockpile model was set to use $\lceil \mathcal{D} \rceil$ pivots selected by the `K-MEDOIDS` strategy [Kaufman and Rousseeuw 1987], CDH's constrained by 128 buckets, and to provide *pessimistic estimations* for both range and k-NN queries. By using a fixed budget of memory, we evaluate our model against two baseline competitors, as follows:

1. The model in [Ciaccia et al. 1998], which was set to use a single Equi-Width histogram constrained by 256 buckets as the representation of the pairwise distance distribution, and
2. The model in [Baioco et al. 2007], which was set to use the fractal dimension drawn from the Distance-Plot graphs.

The compared methods were implemented under the same framework by using the Arboretum library[5], the g++ 4.9.2 compiler and the Ubuntu 14.04 OS running on an Intel Core i7 2.67 GHz, 6 GB of RAM and HDD SATA III 7200 RPM. Additionally, the accuracy of the evaluated models was normalized in terms of the absolute error to the real cost of the searches in all experiments. The range and k-NN costs were collected following the range search procedure in [Ciaccia et al. 1997] and the k-NN search algorithm in [Hjaltason and Samet 2003], respectively.

[2]Available at: `www.ibge.gov.br`
[3]Available at: `www.archive.ics.uci.edu/ml`
[4]Available at: `www.commons.wikimedia.org/wiki/Category:Paintings`
[5]Available at: `www.bitbucket.org/gbdi/arboretum`

Table 1. Data sources and parameters employed in the experiments.

Source	Dim.	δ	$\lceil \mathcal{D} \rceil$	Card.	Description
`CITIES`	2	L_2	2	$5,507$	Geographical coordinates of 5507 Brazilian cities.
`WINE`	11	L_1	6	$6,497$	UCI labeled data source of wine's description.
`LETTER`	16	L_∞	9	$20,000$	Statistical moments and edge counts from all letters of the A–Z alphabet.
`CANVAS`	16	L_2	6	$3,879$	MPEG-7 Color Layout features from Wikimedia paintings.

5. Comparison of range queries cost estimates

We employed the 10-folds cross validation approach (90% of data for indexing, 10% of data for querying, cycling) for the evaluation of Stockpile and range queries. For each data source, we took the maximum distance between a pair of elements (`max`) and define 6 thresholds (ξ) for the range queries. Basically, we set ξ as a percentage of `max` from 5% to 30% in steps of 5%. Figure 3 shows the comparison between evaluated models. Each point in Figure 3(a) represents the absolute difference between the predicted and real number of scanned nodes when executing a range query. Stockpile achieved up to 36%, 31%, 41% and 54% more accurate disk accesses' predictions in comparison to model in [Ciaccia et al. 1998] regarding data sources `CITIES`, `WINE`, `LETTER` and `CANVAS`, respectively. Moreover, our approach reached up to 85%, 78%, 62% and 81% better predictions than model in [Baioco et al. 2007] for the same data sources. Despite the behaviors vary for increasing values of ξ, Stockpile was particularly dominant for smaller ξ values, which are the most meaningful ones for the majority of applications. Similar results were obtained regarding the predicted number of distance calculations, as shown in Figure 3(b). Again, Stockpile was up to 51%, 49%, 11% and 23% more accurate than model in [Ciaccia et al. 1998] for data sources `CITIES`, `WINE`, `LETTER` and `CANVAS`, respectively. Our model also was 64%, 73%, 25% and 69% better than model in [Baioco et al. 2007] in the same scenario.

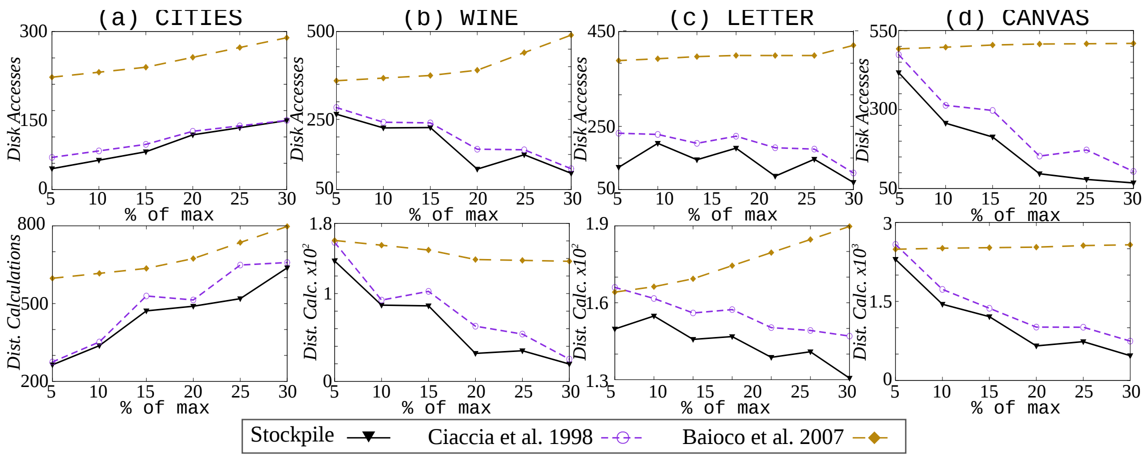

Figure 3. Comparison of models for range queries from 5% to 30% of `max`.

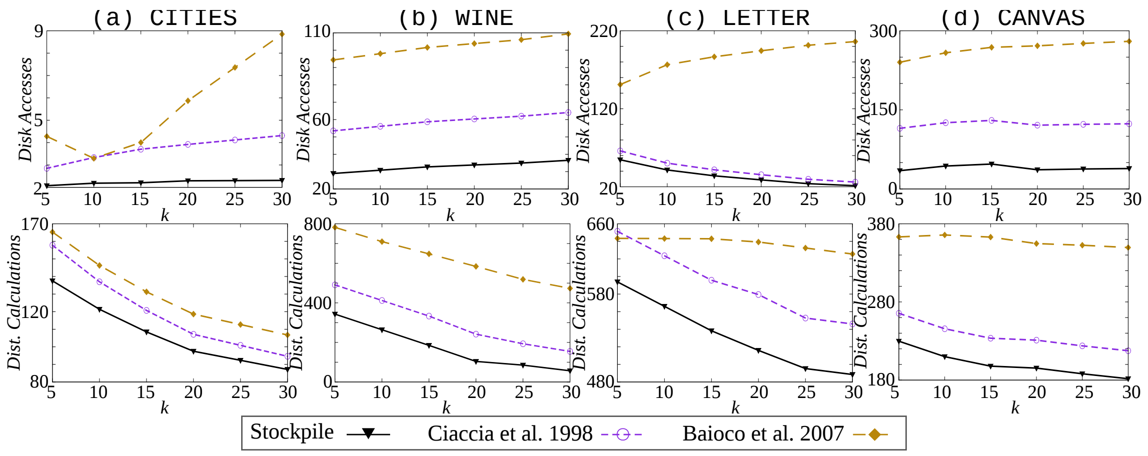

Figure 4. Comparison of models for k-NN queries from $k = 5$ to 30 in steps of 5.

6. Comparison of k-NN queries cost estimates

We also employed the 10-folds cross validation procedure for the evaluation of Stockpile accuracy in the task of predicting the cost of k-NN queries. In the experiment, we vary the parameter k from 5 to 50 in steps of 5 and evaluated our approach on terms of disk accesses and distance calculations. Figure 4(a) shows the Stockpile was up 46%, 43%, 19% and 54% more precise than model in [Ciaccia et al. 1998] for the estimation of disk accesses of k-NN queries on data sources `CITIES`, `WINE`, `LETTER` and `CANVAS`, respectively. Our model also was up to 73%, 69%, 85% and 83% better than model in [Baioco et al. 2007] in the same evaluation. Moreover, Figure 4(b) shows Stockpile was up to 13%, 30%, 13% and 12% more precise than the approach in [Ciaccia et al. 1998] in the estimation of distance calculations of k-NN queries on data sources `CITIES`, `WINE`, `LETTER` and `CANVAS`. Additionally, our approach was up to 20%, 80%, 26% and 45% better than model in [Baioco et al. 2007] in the same scenario. We highlight Stockpile was the most accurate in both range and k-NN queries. For instance, Stockpile not only outperformed its competitors for smaller radii (k-NN for $k \leq 30$), but it was also the best choice for larger values of radii (range for $\xi > 5\%$ of `max`). Accordingly, we point out the predictions drawn from pivot-based histograms are versatile enough to reach a good precision in comparison to estimates drawn from single pairwise distance distributions.

7. Conclusions

Cost modeling of similarity searches requires the proper handling of distance distributions. In this study, we proposed the Stockpile cost model that estimates the cost of similarity queries according to their locality by using pivot-based distance histograms. We compared Stockpile to strategies in [Ciaccia et al. 1998] and [Baioco et al. 2007] in the evaluation of similarity queries and the results showed our model was up to 54% and 85% more accurate than these two competitors, respectively. Future works include the evaluation of our model on other data sources and metric trees to verify the effects of the Stockpile parameters (e.g. number of pivots and optimistic/pessimistic estimates) regarding distinct ball-partitioning strategies.

Acknowledgments. The authors thank CNPq, CAPES, and FAPESP for their financial support.

References

Aly, A. M., Aref, W. G., and Ouzzani, M. (2015). Cost estimation of spatial k-nearest-neighbor operators. In *EDBT*, pages 457–468.

Baioco, G. B., Traina, A. J. M., and Traina Jr., C. (2007). MAMCost: Global and Local Estimates leading to Robust Cost Estimation. In *SSDBM*, pages 1:1–1:6.

Bedo, M. V. N., Kaster, D. S., Traina, A. J. M., and Traina Jr., C. (2015). CDH: a novel structure to boost k-nearest neighbor queries. In *SSDBM*, pages 36:1–36:6.

Bustos, B., Navarro, G., and Chávez, E. (2003). Pivot selection techniques for proximity searching in metric spaces. *PRL*, 24(14):2357–2366.

Ciaccia, P., Patella, M., and Zezula, P. (1997). M-tree: An efficient access method for similarity search in metric spaces. In *VLDB*, pages 426–435.

Ciaccia, P., Patella, M., and Zezula, P. (1998). A Cost Model for Similarity Queries in Metric Spaces. In *PODS*, pages 59–68.

Hjaltason, G. R. and Samet, H. (2003). Index-driven similarity search in metric spaces. *TDS*, 28(1):517–580.

Ioannidis, Y. (2003). The history of histograms (abridged). In *VLDB*, pages 19–30.

Kaufman, L. and Rousseeuw, P. (1987). *Clustering by means of medoids*. North-Holland.

König, A. C. and Weikum, G. (2002). A framework for the physical design problem for data synopses. In *EDBT*, pages 627–645.

Korn, F., Pagel, B., and Faloutsos, C. (2001). On the 'Dimensionality Curse' and the 'Self-Similarity Blessing'. *TKDE*, 13(1):96–111.

Lokoč, J. (2010). *Tree-based indexing methods for similarity search in metric and non-metric spaces*. PhD thesis, Charles University, Ovocný trh 3-5, 116 36 Praha.

Lu, Y., Lu, J., Cong, G., Wu, W., and Shahabi, C. (2014). Efficient algorithms and cost models for reverse spatial-keyword kNN search. *TDS*, 39(2):13:1–13:46.

Navarro, G., Paredes, R., Reyes, N., and Bustos, C. (2017). An empirical evaluation of intrinsic dimension estimators. *IS*, 64:206–218.

Skopal, T., Pokorný, J., and Snásel, V. (2004). PM-tree: Pivoting Metric Tree for Similarity Search in Multimedia Databases. In *ADBIS*, pages 1 – 16.

Tao, Y., Zhang, J., P., D., and Mamoulis, N. (2004). An efficient model for optimization of kNN in low and medium dimensional spaces. *TKDE*, 16(10):1169–1184.

Tasan, M. and Özsoyoglu, Z. M. (2004). Improvements in distance-based indexing. In *SSDBM*, pages 161–170.

Traina Jr., C., Traina, A. J. M., Faloutsos, C., and Seeger, B. (2002). Fast indexing and visualization of metric data sets using slim-trees. *TKDE*, 14(2):244–260.

Vieira, M. R., Traina Jr., C., Traina, A. J. M., Arantes, A. S., and Faloutsos, C. (2007). Boosting kNN queries estimating suitable query radii. In *SSDBM*, pages 1:1 – 1:10.

Zezula, P., Amato, G., Dohnal, V., and Batko, M. (2006). *Similarity Search - The Metric Space Approach*, volume 32. Kluwer.