A new approach for similarity queries using proximity graphs

Alexander Ocsa[1], **Carlos Bedregal**[2], **Ernesto Cuadros-Vargas**[2,3]

[1] Department of Computer Science
National University of San Agustin – Arequipa, Peru

[2]Department of Computer Science
Catholic University of San Pablo – Arequipa, Peru

[3]Peruvian Computer Society

{aocsa,cbedregal}@ieee.org, ecuadros@spc.org.pe

Abstract. *Proximity graphs have been widely used in the area of computational geometry; through a vicinity concept these graphs establish relations of similarity between elements of a set. In this paper we propose the use of Relative Neighborhood Graph (RNG) in metric spaces in order to efficiently answer similarity queries. Additionally we introduce a new algorithm for range queries and nearest neighbor queries making use of the spatial approximation in graphs. Experiments show that our proposal has a comparable performance in terms of Number of Distance Calculations (NDC) and time.*

1. Introduction

Information queries are an important process in data management, even more nowadays with the increasing availability of information in various forms, such as video sequences, images or DNA sequences. Exact search has been the traditional approach for information retrieval; the problem here is that a total order relation is needed among the indexed data. Considering that similarity is the instinctive criterion by which people make comparisons, the information retrieval communities use similarity in order to organize and search for data.

Similarity searches have been widely used in many areas of computer science, such as data mining, bioinformatics and video compression. Metric Access Methods (MAM) have proven excellent to solve similarity searches since they are designed to work over metric spaces reducing the cost of search. Since the distance function can have a high computational cost, it is assumed that the performance of MAM depends mainly on the Number of Distance Calculations (NDC) performed during construction and search processes. Working with large datasets (i.e. multimedia data) the access to secondary memory must also be considered.

Within computational geometry, graph-based methods build proximity graphs, also called neighborhood graphs, on a set of points, creating edges among these points based on a neighborhood criterion. This approach produced excellent results on clustering tasks and on applications of computational vision and pattern recognition, visual perception, biology, geography and cartography. Neighborhood can be defined in terms of geometric attributes such as coordinates, distance, density

or forms. In proximity graphs, Relative Neighborhood Graph (RNG) is a prominent representative [Jaromczyk and Toussaint 1992].

The paper is organized as follows: Section 2 presents a review of previous work. Section 3 describes the proposed technique. Section 4 shows the experimental results. Finally, Section 5 presents the conclusions of the paper.

2. Previous work

A common characteristic in applications of similarity information retrieval is the existence of a universe of objects $\mathbb{U}$ and a function $d : \mathbb{U} \times \mathbb{U} \rightarrow \mathbf{R}$ that measures the distance between two objects of $\mathbb{U}$. Working in metric spaces, we define $\mathbb{S} \subseteq \mathbb{U}$ as a finite set of data that can be preprocessed, where the function $d()$ measures the dissimilarity among objects and satisfies the following conditions, $\forall x, y, z \in \mathbb{U}$:

1. $d(x, y) \geq 0$	positivity
2. $d(x, y) = d(y, x)$	symmetry
3. $d(x, y) = 0 \leftrightarrow x = y$	reflexibility
4. $d(x, y) \leq d(x, z) + d(y, z)$	triangle inequality

Triangle inequality is the most important property because it gives bounds on a distance we may not have computed, leading to similarity search algorithms significantly faster i.e. if we have the distances $d(x, z)$ and $d(y, z)$, the bounds for the unknown distance $d(x, y)$ are $|d(x, z) - d(y, z)| \leq d(x, y) \leq d(x, z) + d(y, z)$ [Clarkson 2006]. That is to say, it is possible to discard regions which do not overlap the query region, as it can be seen in Figure 1.

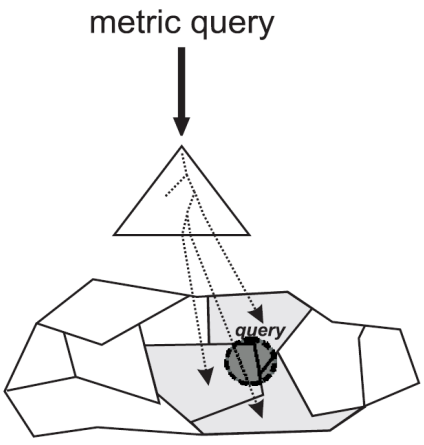


Figure 1. Unified model for searching in metric spaces [Chávez et al. 2001].

Given a query object $q \in \mathbb{U}$, in order to recover similar objects to q, the following basic types of queries are defined:

- **Range query** $Rq(q, r)$: Which finds all elements that are within the query radius r. $Rq(q, r) = \{u \in \mathbb{U} | d(u, q) \leq r\}$.
- **k-Nearest neighbor query** $kNN(q, k)$**:** Which finds the k nearest neighbors of q. Finds a set $A \subseteq \mathbb{U}$ so that $|A| = k$ and $\forall u \in A, v \in U - A, d(q, u) \leq d(q, v)$.

- **Nearest neighbor query** $NN(q)$**:** Which finds the nearest neighbor of q. $NN(q) = kNN(q, 1)$.

Metric Access Methods (MAM) organize a dataset using a similarity criterion. MAMs are structures that work over metric spaces, organizing data for an efficient answer to similarity queries. Good surveys on MAMs can be found in [Chávez et al. 2001], [Hjaltason and Samet 2003], and [Clarkson 2006]. According to [Zezula et al. 2006], MAMs can be classified as:

- Ball-partitioning methods: Fixed Queries Tree, Vantage Point Tree.
- Hyper-plane partitioning methods: Generalized Hyper-plane Tree.
- Precomputed distances: Approximating Eliminating Search Algorithm (AESA).
- Hybrid methods: Multi Vantage Point Tree, GNAT, Spacial Approximation Tree (SA-Tree).
- Others: M-Tree, D-Index.

In [Chávez et al. 2001] M-Tree [Ciaccia et al. 1997] is classified as a method based on covering radius. Thus, Slim-Tree [Caetano Traina et al. 2000] and DBM-Tree [Marcos R. Viera and Traina 2004a] could be also considered under this classification since they work in a similar way.

The M-Tree [Ciaccia et al. 1997] based techniques, grow bottom-up from leaves to rootfrom leaves to the root organizing the objects in a hierarchical structure using a representative as the center of each region which covers the objects in a sub-tree.

The Slim-Tree [Caetano Traina et al. 2000] introduces new split and insertion algorithms as well as the Slim-down algorithm and an overlapping measure called Fat-factor to build faster trees. The split algorithm is based on the Minimum Spanning Tree (MST). The Slim-down algorithm is presented to reduce the degree of overlap, making the metric tree tighter thus improving query and building performance.

The DBM-Tree [Marcos R. Viera and Traina 2004a] operates in a similar way to Slim-Tree. This MAM proposes to relax the height of the tree in regions with high density of data in order to minimize the overlap of nodes. This approach reduces the number of distance computations without affecting the number of disk accesses.

The effort of reducing the overlap makes these techniques to work well in low and medium dimensionality, but working with high dimensional data, the overlapping among regions could be so high that the idea of data representation with a hyper-spherical regions hierarchy deteriorates the results compared with a sequential search [Böhm et al. 2001].

On the other hand, in the field of computational geometry, a graph is defined by $G(V, E)$ where V is a set of points in R^n and $E : V \times V$ a set of edges. In proximity graphs a property P (neighborhood criterion) is defined, and each pair of points $(p, q) \in V \times V$ is connected by an edge $e \in E$ with weight $d(p, q)$ if and only if they fulfill P; then p is neighbor of q and vice versa. Furthermore, proximity graphs connect spatially near points, reflecting the proximity between them.

In order to define the some proximity graphs we must consider: $d : V \times V \rightarrow \mathbf{R}$ a distance function defined in some metric, and $B(x, r)$ a sphere with center in $x \in V$ and radio r, i.e. $B(x, r) = y : d(x, y) < r$. Considering these definitions, some proximity graphs include:

- Delaunay Triangulation (DT): For each triangulation T in V, the circumcircle $C(T)$ does not contain points of V.
- Gabriel Graph (GG): $(x, y) \in E : B(\frac{x+y}{2}, \frac{d(x,y)}{2}) \cap V = \emptyset$.
- RNG: $(x, y) \in E : (B(x, d(x, y)) \cap B(y, d(x, y))) \cap V = \emptyset$. Figure 2 shows the intersection of the two spheres. That is, two points $x, y \in V$ are neighbors if $d(x, y) \leq max\{d(x, z), d(y, z)\} \forall z \in V \wedge z \neq x, y$.
- MST: In a weighted graph, MST is the minimum-weight tree which contains all of the graph's vertices.

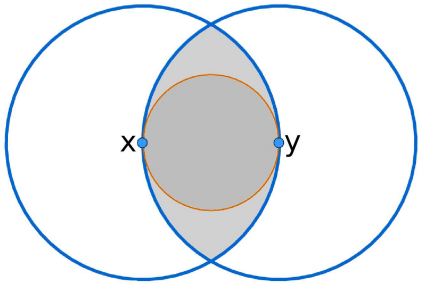


Figure 2. The inner circle shows the exclusion area applied in GG. The gray area shows the exclusion area applied in RNG. In order to connect x and y, there are no other points of V inside the exclusion area.

In [Toussaint 1980] the hierarchy relation $MST \subseteq RNG \subseteq DT$ was demonstrated. In [Toussaint 1980] two construction algorithms for RNG with costs $O(n^3)$ and $O(n^2)$ were also presented. [Supowit 1983] showed construction algorithms with cost $O(nlogn)$. Most of these algorithms work over Euclidean spaces because they are based on the Delaunay Triangulation (DT). Since RNG is defined purely in terms of the metric distance, a construction algorithm not based on DT could be developed using a divide-and-conquer approach, therefore applicable to any metric space.

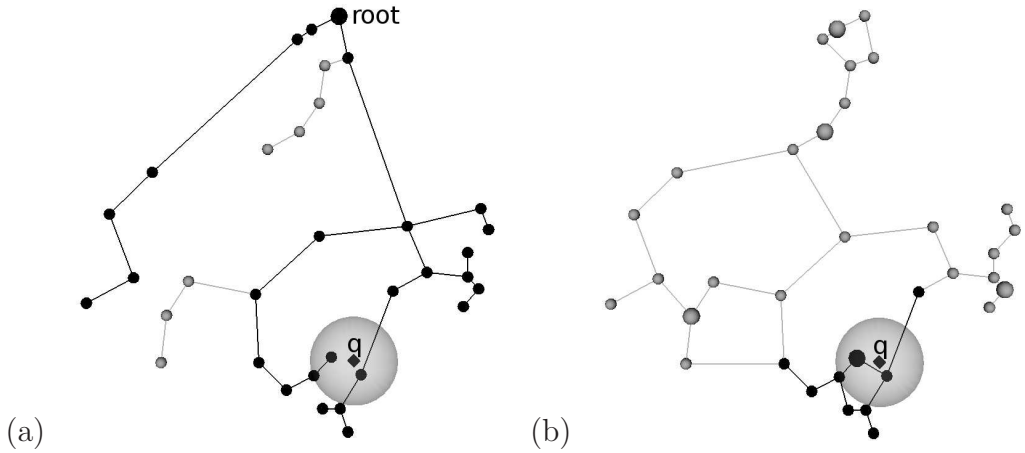
3. Neighborhood Approximation Graph (NA-Graph)

According to the Spatial Approximation Approach [Navarro 2002] starting from $a \in S$, a random element of the graph associated with its set of neighbors $N(a)$, we can spatially approximate towards a query point $q \in \mathbb{U}$. This is done selecting the element $b \in N(a)$ that is closer to q, until reaching the element of S that is the closest to the query.

As it was stated in [Navarro 2002], in order to satisfy the spatial approximation criterion, a graph must fulfill the following condition:

$$\forall a \in S, \forall q \in \mathbb{U}, if\ \forall b \in N(a), d(q, a) \leq d(q, b), then\ \forall b \in S, d(q, a) \leq d(q, b)$$

Although the graph generated by DT fulfills this condition [Navarro 2002], the problem is that it is only true for Euclidean spaces. In general, for metric spaces a completely connected graph is necessary. SA-Tree [Navarro 2002] simplifies the idea of spatial approach in order to find an optimal solution. Here an object a is selected as the root node and connected to objects $s \in S$ such that they are closer to a than to any neighbor (an example is showed in Figure 3(a)). Each object connected to a is a root of an appropriate subtree. A static approach was originally designed for SA-Tree, and dynamic version of SA-Tree was later proposed in [Navarro and Reyes 2002]. A drawback of SA-Tree is that the selection of the root is critical for the performance of the search algorithm; furthermore a bad root could lead to the exploration of most part of the nodes of the tree as it can be observed in Figure 3(a).



(a) (b)

Figure 3. Example of range query with the SAT (a) and NA-Graph (b). In (a) the root, and in (b) the seeds have a greater node size. The dataset explored is shaded black, and the gray sphere represents the query region.

Since RNG is a subgraph of DT in which it is guaranteed that for any pair of points $a, q \in S$, exists one or more paths where starting at a we can arrive at q, it is possible to spatially approximate towards a query point following the path of minimum length within RNG as it is observed in Figure 4. This investigation demonstrates that RNG can also be useful as a data structure that answers similarity queries using a spatial approximation criterion .

For Nearest Neighbor (NN) queries in RNG, we start from a node a of the graph, and in order to spatially approximate to a query point $q \in \mathbb{U}$, we move to the node $b \in N(a)$ which is closest to q until we have arrived at a node already explored (entered into a loop). If the node a is closer to q than b, then a is marked as a candidate to be NN of q; after finishing the exploration, the closer candidate is chosen as the nearest neighbor of q. The process is detailed in Algorithm 1 and in Figure 5.

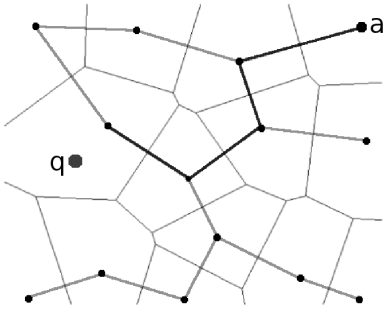


Figure 4. Starting at point a we move towards q. The corresponding Voronoi partition is demarcated in dashed lines.

Algorithm 1 NearestNeighbor(Query q)

1: Select a point $a \in S$;
2: **while** a has not been explored **do**
3: Select the node $b \in N(a)$ which is the closest one to q;
4: **if** $\forall b \in N(a)$, $d(a,q) < d(b,q)$ **then**
5: Mark a as candidate;
6: **end if**
7: $a \leftarrow b$;
8: **end while**
9: Choose the candidate which is the closest one to q.

end

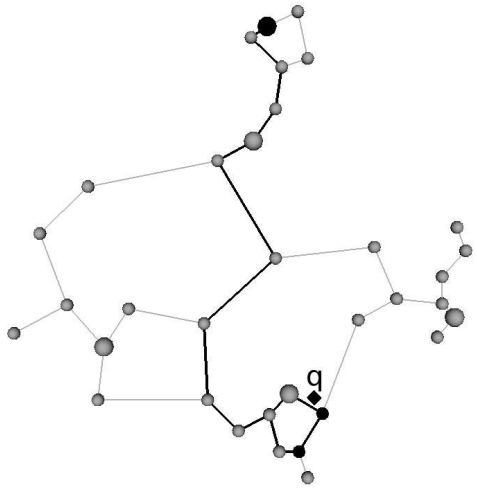


Figure 5. An example of NN search in NA-Graph. In the graph the seeds have a greater node size. In this case a random seed was chosen as the starting search point. The path explored is shaded black as well as the candidate nodes.

The construction process of this structure is the same as the RNG, as it was seen on the previous section: time of construction depends on the algorithm we use.

Once the capacity of a spatial approximation in RNG is demonstrated, this idea can be generalized to solve similarity queries as stated in [Navarro 2002].

To solve range queries in RNG: given a query point q with radius r, starting from a node a, explore recursively the elements $b \in N(a)$ whose distance to q is smaller or equal than $d(x,q)+2r$, where x is the closest node to q already explored. This process is repeated until entering into a loop. Finally all the explored nodes within the search radius are reported. The process is detailed in Algorithm 2 and in Figure 3 (b). Note that It is first invoked as $RangeQuery(a, q, r, d(a, q))$, where a is the seed of the graph.

Algorithm 2 RangeQuery (Point a, Query q, Range r, Distance mindist)

```
1: if a has not been explored then
2:   if d(a,q) ≤ r then
3:     Insert a to query Result
4:   end if
5:   mindist ← min{{mindist} ∪ {d(c,q) : c ∈ N(a)}}
6:   for b ∈ N(a) do
7:     if d(b,q) ≤ mindist + 2r then
8:       RangeQuery(b, q, r, mindist);
9:     end if
10:  end for
11: end if
```

end

Given a query point q, in order to perform a similarity query in RNG, a seed or starting point $a \in S$ could be randomly selected. If a is located far from q, it is likely that most of the graph would be explored; therefore, it is of interest that the seed would be located as close to q as possible.

We propose the creation of the set of seeds $Seeds(S) \subset S$ that groups a few representative nodes of RNG, selected based on a density criterion. Then the node $s \in Seeds(S)$ that is closer to the query is chosen as the seed. The density criterion Relative Neighborhood Density Factor (RNDF) is a value based on the neighborhood of each node of RNG. RNDF is defined in Equation 1.

$$RNDF(x) = \frac{|\{y \in N(x) : x \leftarrow NN(y)\}|}{|N(x)|}, \forall x \in S \tag{1}$$

According to Equation 1, $RNFD(x)$ has value 1 only if all the neighbors of x have x as their nearest neighbor; in any other case this factor has a value smaller than 1. Therefore, we are interested in select as possible seeds those points with the highest density factor. In order to reduce the number of candidates to seed, we discard candidates with one neighbor and candidates that have a neighbor that already was selected as candidate. Equation 2 defines the set $Seeds(S)$.

$$Seeds(S) = \{x \in S : RNDF(x) \geq 1 \wedge |N(x)| > 1 \wedge N(x) \cap Seeds(S) = \emptyset\} \tag{2}$$

Note that it is possible to index the set $Seeds(S)$ with a Metric Access Methods (MAM) in order to accelerate the process of searching the nearest seed to q.

4. Experiments

Based on this idea, three groups of experiments are presented using the SA-Tree [Navarro 2002], Slim-Tree [Caetano Traina et al. 2000] and DBM-Tree [Marcos R. Viera and Traina 2004b] and our proposal Neighborhood Approximation Graph (NA-Graph). In the first group of experiments the range query process of our approach is compared the mentioned MAMs, each technique measured in terms of Number of Distance Calculations (NDC). The second group of experiments compares the proposed index to the same MAMs in terms of the NDC performed in k-NN queries. Finally we compare visually the resulting graph of NA-Graph and the resulting tree of SA-Tree when performing range queries. All experiments were implemented using Microsoft Visual C++ 6.0 on a PC 2.0 GHz, 1024 MB of RAM, running Microsoft Windows XP.

Three datasets were used for our experiments:

- CITIES: This dataset contains a set of 5,507 geographical coordinates of the Brazilian cities. The file was obtained from `http://www.ibge.gov.br`.
- SYNT2D: This dataset contains 1000 2-d synthetic points between $(0, 0)...(5, 5)$. We choose this dataset in order to have a visualization of the search process.
- SYNT16D: This dataset contains 10,000 16-d synthetic points between $(0, 0)...(5, 5)$.

From each dataset, were extracted randomly 500 objects to be used as query centers. In the query measurements, each point corresponds to performing 500 queries with the same parameters but varying query centers.

Figures 6(a), 6(c) and 6(e) illustrate the first group of experiments. Here we make a comparison in terms of the accumulated Number of Distance Calculations (NDC) when performing range queries.

As it can be observed in the first group of experiments, NA-Graph reduces by up to 40% the NDC in comparison to SA-Tree. This difference is due to the use of seeds that are closer to the query point.

Results for the second group of experiments are presented in Figures 6(b), 6(d) and 6(f). These experiments compare the average Number of Distance Calculations (NDC) when performing kNN queries.

DBM-tree and Slim-Tree are MAMs based on covering radius. To discard a region (a node) we only verify the overlap between the query region and the node. For this reason these methods work well in low and medium dimensionality, but when working with high dimensional data, the overlapping among region could be high, so deteriorates the result.

On the other hand, SA-Tree and NAG are methods based on spatial approximation criterion, that is dividing the search space, starting at some point in the space and get closer and closer to the query. To discard a element is necessary to verify that the point is outside the region $(min\{d(x, q)\} + 2 * r)$. So NA-Graph give better performance that SAT due to the use of seeds that are closer to the query point, that is find the value $min\{d(x, q)\}$ faster, also we show that NA-Graph is

competitive against methods based on covering radius. The SA-tree gives less Number of Distance Calculations (NDC) than the other existing structures on metric spaces of high dimension or queries with low selectivity but in low dimensions or for queries with high selectivity (small r or k), its search performance is poor.

Finally, Figures 7 and 8 show SA-Tree and NA-Graph respectively using the SYNT2D dataset in the worst range query case. Figures demonstrate that the area explored in NA-Graph is less than SA-Tree.

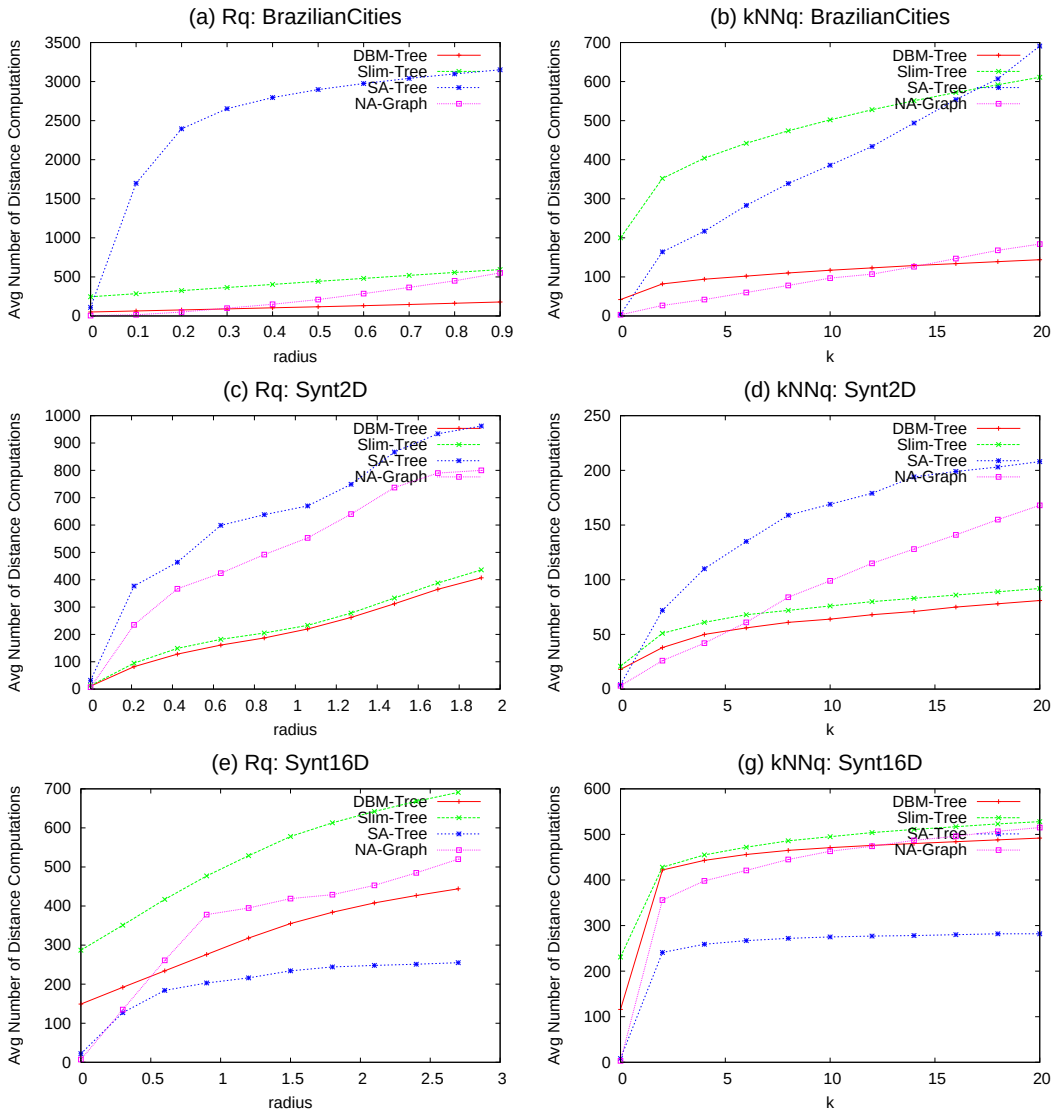


Figure 6. Comparison of the average number of distance calculations in the range queries (first column), and the average number of distance calculations in the k-Nearest Neighbor queries (second column) of DBM-Tree, Slim-Tree, SA-Tree and NA-Graph for the $BrazilianCities$ ((a) - rq and (b) - $kNNq$), $Synt2D$ ((c) - rq and (d) - $kNNq$) and $Synt16D$ ((e) - rq and (f) - $kNNq$) datasets.

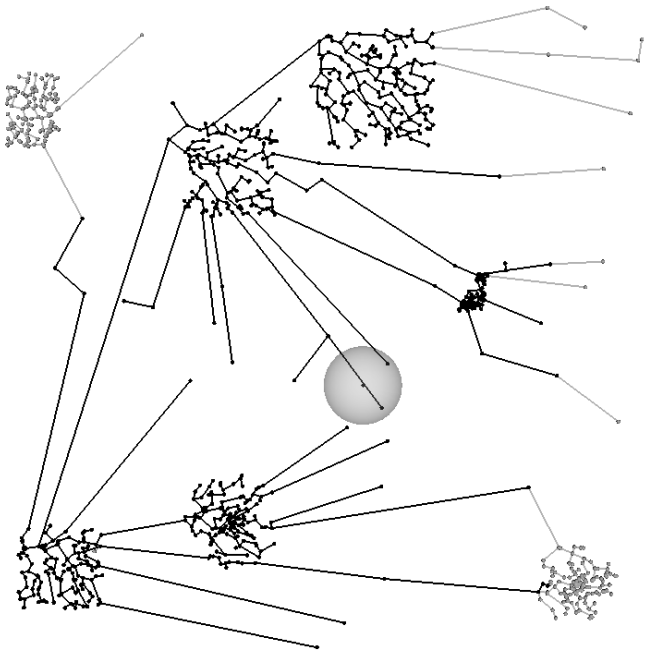


Figure 7. Visualization of worst range query case in SA-Tree using the SYNT2D dataset.

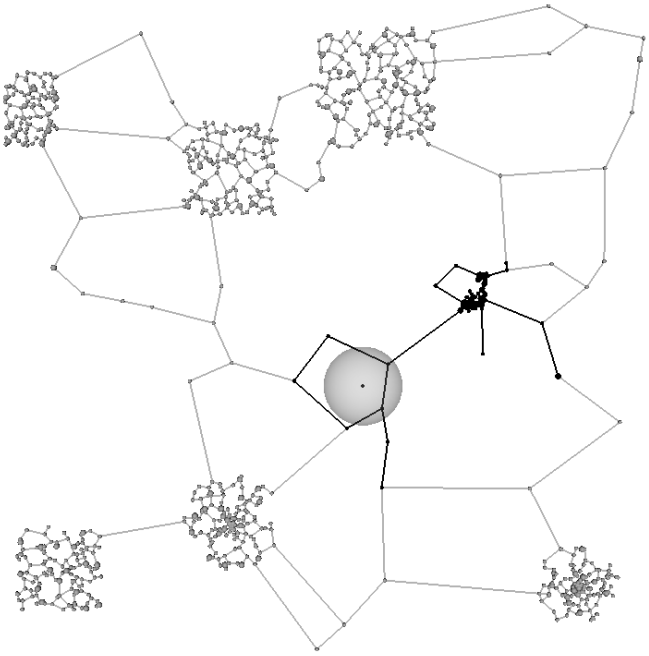


Figure 8. Visualization of worst range query case in NA-Graph using the SYNT2D dataset.

5. Conclusions

The experiments showed that NA-Graph can be used as MAM based on the spatial approximation approach. Performance of NA-Graph was up to 40% better than SA-Tree. In general NA-Graph is competitive with MAMs based on covering radius.

Through Relative Neighborhood Density Factor (RNDF) we select a subset of representative nodes in order to reduce the graph exploration, thus speeding up similarity queries.

This is the first approach that introduces Relative Neighborhood Graph (RNG) to solve similarity queries. Furthermore, NA-Graph have a large potential for improvements we are pursuing. First develop a construction algorithm using RNG as data structure based on clustering to improve even more the result. Second new search algorithms using neighborhood information.

References

Böhm, C., Berchtold, S., and Keim, D. A. (2001). Searching in high-dimensional spaces: Index structures for improving the performance of multimedia databases. *ACM Comput. Surv.*, 33(3):322–373.

Caetano Traina, J., Traina, A. J. M., Seeger, B., and Faloutsos, C. (2000). Slim-trees: High performance metric trees minimizing overlap between nodes. In *Proc. of the 7th International Conference on Extending Database Technology EDBT00*, pages 51–65, London, UK. Springer-Verlag.

Chávez, E., Navarro, G., Baeza-Yates, R., and Marroquín, J. L. (2001). Searching in metric spaces. ACM Press.

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

Clarkson, K. L. (2006). Nearest-neighbor searching and metric space dimensions. In Shakhnarovich, G., Darrell, T., and Indyk, P., editors, *Nearest-Neighbor Methods for Learning and Vision: Theory and Practice*, pages 15–59. MIT Press.

Hjaltason, G. R. and Samet, H. (2003). Index-driven similarity search in metric spaces. *ACM Trans. Database Syst.*, 28(4):517–580.

Jaromczyk, J. and Toussaint, G. (1992). Relative neighborhood graphs and their relatives. *P-IEEE*, 80:1502–1517.

Marcos R. Viera, Caetano Traina Fr., F. J. T. C. and Traina, A. J. (2004a). DBM-tree: A dynamic metric access method sensitive to local density data. Brazilian Symposium on Databases.

Marcos R. Viera, Caetano Traina Fr., F. J. T. C. and Traina, A. J. (2004b). DBM-tree: A dynamic metric access method sensitive to local density data. Brazilian Symposium on Databases.

Navarro, G. (2002). Searching in metric spaces by spatial approximation. *The VLDB Journal*, 11(1):28–46.

Navarro, G. and Reyes, N. (2002). Fully dynamic spatial approximation trees.

Supowit, K. J. (1983). The relative neighborhood graph, with an application to minimum spanning trees. *J. ACM*, 30(3):428–448.

Toussaint, G. T. (1980). The relative neighborhood graph of a finite planar set. *Pattern Recognition*, 12:261–268.

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