Journal of Computer Applications ISSN: 0974 – 1925,
Volume-5, Issue EICA2012-3, February 10, 2012

Efficient Ranking on Road Network by Quality Preferences

Akila G
Final ME (CSE),
Muthayammal Engineering College,
Rasipuram, India.
E-mail: akilapraneet@gmail.com

Sumathi G
Lecturer (CSE),
Muthayammal Engineering College,
Rasipuram, India.
E-mail: subamecsep@gmail.com

Abstract — A spatial preference query ranks objects based on the qualities of features in their spatial neighborhood. For example, consider a road network database with available paths for two points. A customer may want to rank the paths with respect to their distance defined after aggregating the qualities of other features (road condition, travelling time, traffic monitoring and road length) within a distance range. This ranking is obtained by range score and influence score function. Propose appropriate indexing techniques and search algorithms for computing the spatial preference queries on a road network. Extensive evaluation of our methods on both real and synthetic data reveals that an optimized branch-and-bound solution is efficient and robust with respect to different parameters.

Index Terms — *Query processing, spatial databases.*

I. INTRODUCTION

Spatial database systems manage large collections of geographic entities, which apart from spatial attributes contain nonspatial information (e.g., name, size, type, price, etc.). In this paper, we study an interesting type of preference queries, which select the best spatial location with respect to the quality of facilities in its spatial neighborhood.

Given a set D of interesting objects (e.g., candidate locations), a top-k spatial preference query retrieves the k objects in D with the highest scores. The score of an object is defined by the quality of features (e.g., facilities or services) in its spatial neighborhood. As a motivating example, consider a real estate agency office that holds a database with available flats for lease. Here “feature” refers to a class of objects in a spatial map such as specific facilities or services. A customer may want to rank the contents of this database with respect to the quality of their locations,quantified by aggregating nonspatial characteristics of other features (e.g., restaurants, cafes, hospital, market, etc.) in the spatial neighborhood of the flat (defined by a spatial range around it). Quality may be subjective and query-parametric. For example, a user may define quality with respect to non spatial attributes of restaurants around it (e.g., whether they serve seafood, price range, etc.).

As another example, the user (e.g., a tourist) wishes to find a hotel p that is close to a high-quality restaurant and a high quality cafe. Fig. 1a illustrates the locations of an object data set D(hotels) in white, and two feature data sets: the set F1(restaurants) in gray, and the set F2 (cafes) in black. For the ease of discussion, the qualities are normalized to values in [0,1].The score of p of a hotel p is defined in terms of: 1) the maximum quality for each feature in the neighborhood region of p, and 2) the aggregation of those qualities.

A simple score instance, called the range score, binds the neighborhood region to a circular region at p with radius r shown as a circle in Fig.1a, and the aggregate function to SUM. For instance, the maximum quality of gray and black points within the circle of p1 are 0.9 and 0.6, respectively, so the score of p1 is 0.9+0.6=1.5.Similarly, we obtain score of p2 is 1.0+0.1=1.1 and score of p3 is 0.7+0.7=1.4. Hence, the hotel p1 is returned as the top result.

In fact, the semantics of the aggregate function is relevant to the user's query. The SUM function attempts to balance the overall qualities of all features. For the MIN function, the top result becomes p3, with the score of min is 0.7.It ensures that the top result has reasonably high qualities in all features. For the MAX function, the top result is p2, with max{1.0,0.1}=1.0. It is used to optimize the quality in a particular feature, but not necessarily all of them.

The neighborhood region in the above spatial preference query can also be defined by other score functions. A meaningful score function is the influence score. As opposed to the crisp radius r constraint in the range score, the influence score smoothens the effect of r and assigns higher weights to cafes that are closer to the hotel. Fig.1b shows a hotel P_5 and three cafes S_1,S_2,S_3 (with their quality values). The circles have their radii as multiples of r.Now, the score of a café Si is computed by multiplying its quality with the weight 2^{-j} , where j is the order of the smallest circle containing Si. For example, the scores of S_1, S_2,and S_3 are 0.15,0.225, and 0.125 respectively. The influence score of P_5 is taken as the highest value (0.225).

Traditionally, there are two basic ways for ranking objects: 1) spatial ranking, which orders the objects according to their distance from a reference point, and 2) non spatial ranking, which orders the objects by an aggregate function on their non spatial values. Our top-k spatial preference query integrates these two types of ranking in an intuitive way. As indicated by our examples, this new query has a wide range of applications in service recommendation and decision support systems.

To our knowledge, there is no existing efficient solution for processing the top-k spatial preference query. A brute force approach for evaluating it is to compute the scores of all objects in D and select the top-k ones. This method, however, is expected to be very expensive for large input data sets. In this paper, we propose alternative techniques that aim at minimizing the I/O accesses to the object and feature data sets, while being also computationally efficient. Our techniques apply on spatial-partitioning access methods and compute upper score bounds for the objects indexed by them, which are used to effectively prune the search space. Specifically, we contribute the branch-and-bound (BB) algorithm and the feature join (FJ) algorithm for efficiently processing the top-k spatial preference query.

Furthermore, this paper studies top-k spatial preference query on road network and in which the distance between two points defined by their shortest path distance that have not been investigated in our preliminary work[1] and three relevant extensions have been proposed.The first extension is an optimized version of BB that exploits a more efficient technique for computing the scores of the objects. The second extension studies adaptations of the proposed algorithms for aggregate functions other than SUM, e.g., the functions MIN and MAX. The third extension develops solutions for the top-k spatial preference query based on the influence score.

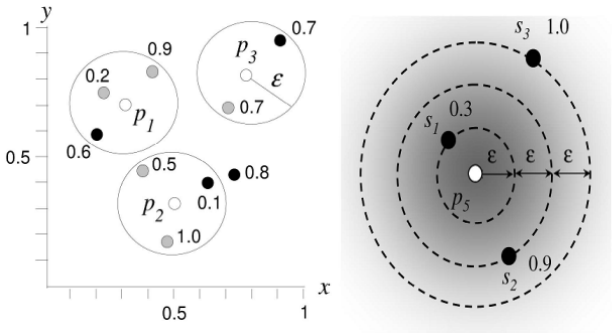

Figure 1: (a) Range score, (b) Influence score, ε =0.2 km

II. LITERATURE REVIEW

Object ranking is a popular retrieval task in various applications. In relational databases, we rank tuples using an aggregate score function on their attribute values [3]. For example, a real estate agency maintains a database that contains information of flats available for rent. A potential customer wishes to view the top 10 flats with the largest sizes and lowest prices. In this case, the score of each flat is expressed by the sum of two qualities: size and price, after normalization to the domain [0,1]. In spatial databases, ranking is often associated to nearest neighbor (NN) retrieval. Given a query location, we are interested in retrieving the set of nearest objects to it that qualify a condition. Assuming that the set of interesting objects is indexed by an R-tree [4], we can apply distance bounds and traverse the index in a branch-and-bound fashion to obtain the answer [5].

Nevertheless, it is not always possible to use multidimensional indexes for top-k retrieval. First, such indexes break down in high-dimensional spaces [6], [7]. Second, top-k queries may involve an arbitrary set of user-specified attributes from possible ones and indexes may not be available for all possible attribute combinations because they are too expensive to create and maintain. Third, information for different rankings to be combined for different attributes could appear in different databases in a distributed database scenario and unified indexes may not exist for them. Solutions for top-k queries [8], [3], [9], [10] focus on the efficient merging of object rankings that may arrive from different sources. Their motivation is to minimize the number of accesses to the input rankings until the objects with the top-k aggregate scores have been identified. To achieve this, upper and lower bounds for the objects seen so far are maintained while scanning the sorted lists.

In the following sections, we first review the R-tree, which is the most popular spatial access method and the NN search algorithm of [5]. Then, survey our feature-based spatial queries.

2.1 Spatial Query Evaluation on R-Trees

The most popular spatial access method is the R-tree [4], which indexes minimum bounding rectangles (MBRs) of objects. Fig. 2 shows a set D ={p1 ... p8} of spatial objects and an R-tree that indexes them. R-trees can efficiently process main spatial query types,

including spatial range queries, nearest neighbor queries, and spatial joins. Given a spatial region W, a spatial range query retrieves from D the objects that intersect W. For instance, consider a range query that asks for all objects within the shaded area in Fig. 2. Starting from the root of the tree, the query is processed by recursively following entries, having MBRs that intersect the query region. For instance, e1 does not intersect the query region, thus the subtree pointed by e1cannot contain any query result. In contrast, e2 is followed by the algorithm and the points in the corresponding node are examined recursively to find the query result p7.

A nearest neighbor query takes as input a query object q and returns the closest object in D to q. For instance, the nearest neighbor of q in Fig. 2 is p7. Its generalization is the k-NN query, which returns the k closest objects to q, given a positive integer k. NN (and k-NN) queries can be efficiently processed using the best-first (BF) algorithm of [5], provided that D is indexed by an R-tree. A min-heap H which organizes R-tree entries based on the (minimum) distance of their MBRs to q is initialized with the root entries. In order to find the NN of q in Fig. 2, BF first inserts to H entries e1, e2, e3, and their distances to q. Then, the nearest entry e2 is retrieved from H and objects p1,p7,p8 are inserted to H. The next nearest entry in H is p7, which is the nearest neighbor of q. In terms of I/O, the BF algorithm is shown to be no worse than any NN algorithm on the same R-tree [5].

The aggregate R-tree (aR-tree) [11] is a variant of the R tree, where each non leaf entry augments an aggregate measure for some attribute value of all points in its sub tree. As an example, the tree shown in Fig. 2 can be upgraded to a MAX aR-tree over the point set, if entries e_1,e_2,e_3 contain the maximum measure values of sets $\{p_2,p_3\}$,$\{p_1,p_8,p_7\}$,$\{p_4,p_5,p_6\}$ respectively. Assume that the measure values of p4,p5,p6 are 0.2, 0.1, 0.4, respectively. In this case, the aggregate measure augmented in e3 would be max{0.2, 0.1, 0.4} = 0.4. In this paper, we employ MAX aR-trees for indexing the feature data sets , in order to accelerate the processing of top-k spatial preference queries.

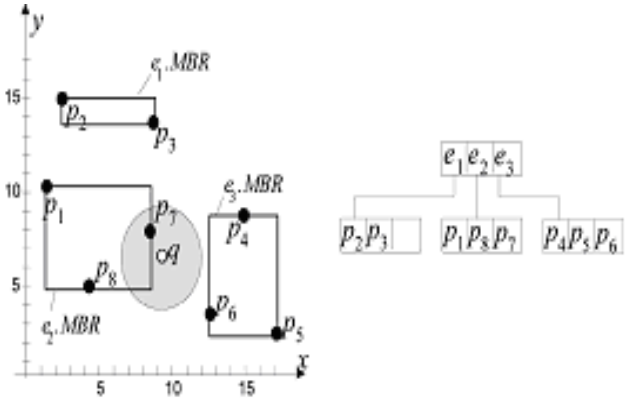

Figure 2: Spatial queries on R-trees.

III. METHODOLOGY

Spatial preference query refers to ranking objects based on quality of features in its spatial neighborhood. Neighborhood concept specified by range score and influence score function. Qualities of features in road network such as road condition, traffic monitoring, road length, and vehicles type are considered. Brute force approach has been used for evaluating spatial preference query in previous work [2]. It computes scores of all objects in given set and select top-k ones and it was very expensive for large input data sets. The proposed work aims to minimize I/O accesses to object and feature data sets, while being computationally efficient and in which the distance between two points defined by their shortest path distance.

Spatial preference query integrates two types of ranking that are Spatial Ranking and Non Spatial Ranking. Spatial Ranking refers to ranking objects based on distance from reference point. Non Spatial Ranking based on aggregated qualities of features in road network.

Apply on spatial-partitioning access methods and compute upper score bounds for the objects indexed in R-tree, used to effectively prune the search space.In this paper, we assume that the object data set D is indexed by an R-tree and each feature data set Fc is indexed by an MAX a R-tree, where each non leaf entry augments the maximum quality of features in its sub tree. The rationale of indexing different feature data sets by separate aR-trees is that: 1) a user queries for only few features (e.g., road condition and road length) out of all possible features (e.g., traffic monitoring, road length, road condition, vehicle types,etc.), and 2) different users may consider different subsets of features.Based on the above indexing scheme, we develop various algorithms for processing top-k spatial preference queries and compute upper bound score computation.

3.1 Group Probing Algorithm

Due to separate score computations for different objects, SP is inefficient for large-object data sets. In view of this, we propose the group probing (GP) algorithm, a variant of SP, that reduces I/O cost by computing scores of objects in the same leaf node of the R-tree concurrently. In GP, when a leaf node is visited, its points are first stored in a set V and then their component scores are computed concurrently at a single traversal of the Fc tree.

Algorithm 1 shows the procedure for computing the cth component score for a group of points. Consider a subset V of D for which we want to compute their range score at feature tree Fc. Initially, the procedure is called with N being the root node of Fc. If e is a non leaf entry and its mindist from some point pεV is within the range ε, then the procedure is applied recursively on the child node of e, since the subtree of Fc rooted at e may contribute to the component score of p. In case e is a leaf entry (i.e., a feature point), the scores of points in V are updated if they are within distance ε from e.

Algorithm 1. Group Range Score Algorithm

algorithm Group Range(Node N, Set V , Value c, Value r)
1: for each entry e ε N do
2: If N is non leaf then
3: If ¥p ε V, min dist (p, e) ≤ r then
4: read the child node N pointed by e;

5: Group Range(N ,V ,c, r);
6: else
7: for each p ε V such that dist(p, e) ≤ r do
8: T(p) = max{ T(p).w(e)};

3.2 Branch-and-Bound Algorithm

GP is still expensive as it examines all objects in D and computes their component scores. We now propose an algorithm that can significantly reduce the number of objects to be examined. The key idea is to compute, for non leaf entries e in the object tree D, an upper bound T(e) of the score T(p) for any point p in the sub tree of e. If T(e) ≤ r then we need not access the sub tree of e, thus we can save numerous score computations.

Algorithm 3. Branch-and-Bound Algorithm

Wk:= new min-heap of size k (initially empty);
V := 0;

algorithm BB(Node N)
1: V:= {e| e ε N};
2: If N is non leaf then
3: for c:= 1 to m do
4: compute T(e) for all e ε V concurrently;
5: remove entries e in V such that T+(e) ≤ v;
6: sort entries e ε V in descending order of T(e);
7: for each entry e ε V such that T(e) > v do
8: read the child node N pointed by e;
9: BB(N);
10: else
11: for c:=1 to m do
12: compute T(e) for all e ε V concurrently;
13: remove entries e in V such that T+(e) ≤ v;
14: update Wk (and v) by entries in V ;

3.2.1 Upper Bound Score Computation

It remains to clarify how the upper bound scores T(e) of non leaf entries within the same node N can be computed concurrently (at Line 4). Our goal is to compute these upper bound scores such that

- the bounds are computed with low I/O cost, and
- the bounds are reasonably tight, in order to facilitate effective pruning.

To achieve this, we utilize only level-1 entries (i.e., lowest level non leaf entries) in Fc for deriving upper bound scores because: 1) there are much fewer level-1 entries than leaf entries (i.e., points), and 2) high-level entries in Fc cannot provide tight bounds.

3.3 Feature Join Algorithm

An alternative method for evaluating a top-k spatial preference query is to perform a multi way spatial join [16] on the feature trees F1, F2, ... , Fm to obtain combinations of feature points which can be in the neighborhood of some object from D. Spatial regions which correspond to combinations of high scores are then examined, in order to find data objects in D having the corresponding feature combination in their neighborhood. In this section, we first introduce the concept of a combination, then discuss the conditions for a combination to be pruned, and finally elaborate the algorithm used to progressively identify the combinations that correspond to query results.

Tuple $(f_1,f_2,\ldots fm)$ is a combination if, for any $c\varepsilon[1,m]$ f_c is an entry (either leaf or nonleaf) in the feature tree Fc. The score of the combination is defined by Σwf_c.

3.4 Modules Description

Spatial Data Events

Spatial database system contains spatial and non spatial information for road network. Select the spatial location according to client preference. Score is defined by the quality of features and features refer to classes of object in spatial map. Quality of the spatial events may be subjective or query parametric. If the spatial events are subjective then the quality with respective to non-spatial attributes, qualities are normalized to values 0 to 1 and quality values can be obtained from rating providers. The Query-parametric Values are based on the queries. Range score binds neighborhood region to crisp radius and the aggregation of qualities. Influence score smoothens the effect of radius and assign the higher weights.

Preferential Queries

The preference queries involve selecting the best spatial location based on multiple feature data sets on road network. It retrieves k points in a data set with highest score. In the preference queries apply the R-tree indexing feature to data sets with three concepts such as MAX aR-tree to road network, efficient tree traversal algorithm and obtain the quality from rating providers .

Ranking of spatial query points

The two basic ways for ranking objects for road network as following: 1.Spatial Ranking **-** It orders according to their distance. 2. Non Spatial Ranking**-** It orders based on aggregate function. By applying the brute-force approach, compute score of all objects in given set and select the top-k ones. It is expensive large data sets. Here the proposal work is to minimize I/O access of features and it is also computationally efficient.

Top-k Spatial Query

Top-k spatial preference query retrieves k objects in database with the highest scores. It uses the concept of Branch-and-bound (BB) algorithm and feature join (FJ) algorithm to compute the upper bound score of objects in optimized way. The solution for top-k queries is obtained via merging of object rankings and minimizes the number of access until top-k aggregates reached. An alternate method for top-k query is multi-way spatial join.

IV. PERFORMANCE EVALUATION

Performance metrics are measured based on the three concepts such as 1. Query size 2. Rank and neighbor range 3. Number of spatial objects.

Performance on Queries with Range Scores

Fig. 3 plots the cost of the algorithms with respect to the number m of feature data sets. The costs of GP, BB, and BB* rise linearly as m increases because the number of component score computations is at most linear to m. On the other hand, the cost of FJ increases significantly with m, because the number of qualified combinations of entries is exponential to
m.

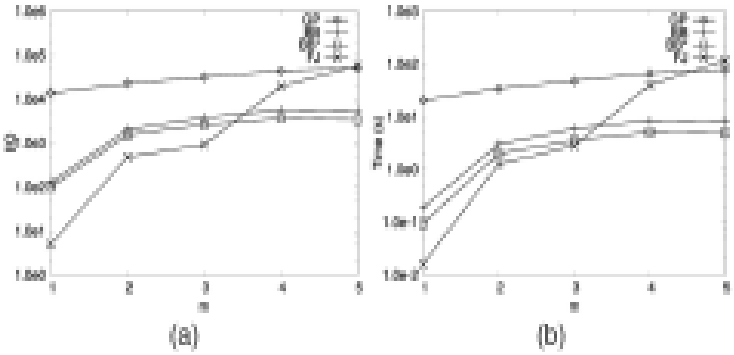

Figure 3: Effect of m, range scores. (a) I/O. (b) Time.

Fig. 4 shows the cost of the algorithms as a function of the number k of requested results. GP, BB, and BB* compute the scores of objects in D in batches, so their performance is insensitive to k. As k increases, FJ has weaker pruning power and its cost increases slightly.

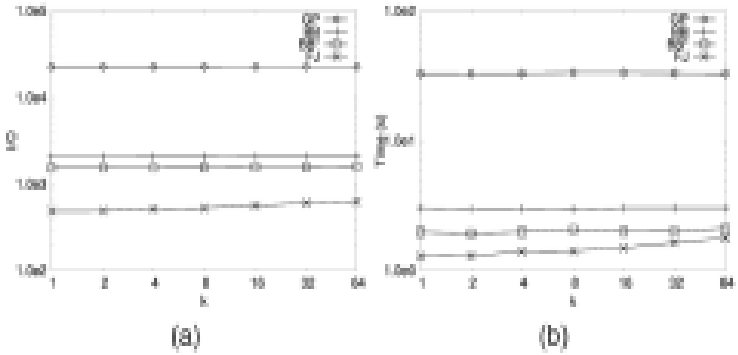

Figure 4: Effect of k, range scores. (a) I/O. (b) Time.

Fig. 5 shows the cost of the algorithms, when varying the query range r. As r increases, all methods access more nodes in feature trees to compute the scores of the points. The difference in execution time between BB* and FJ shrinks as r increases.

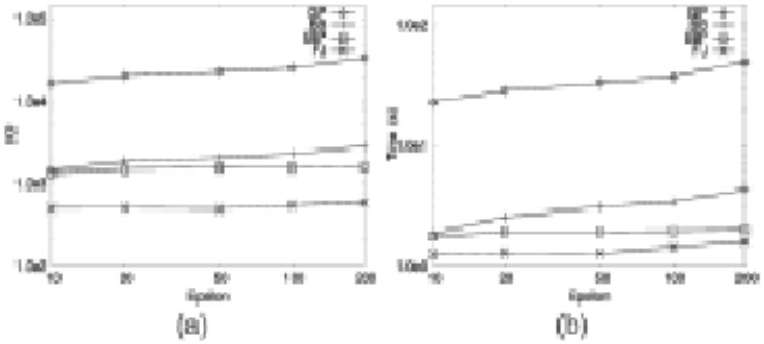

Figure 5: Effect of r, range scores. (a) I/O. (b) Time.

V. CONCLUSION

We studied top-k spatial preference queries, which provide a novel type of ranking for spatial objects based on qualities of features in their neighborhood. The neighborhood of an object p is captured by the scoring function: 1) the range score restricts the neighborhood to a crisp region centered at p, whereas 2) the influence score relaxes the neighborhood to the whole space and assigns higher weights to locations closer to p. We presented few algorithms for processing top-k spatial preference queries. The algorithm GP reduces I/O cost by computing scores of objects in the same leaf node concurrently. The algorithm BB derives upper bound scores for non leaf entries in the object tree, and prunes those that cannot lead to better results. The algorithm BB* is a variant of BB that utilizes an optimized method for computing the scores of objects and upper bound scores of non leaf entries. The algorithm FJ performs a multi way join on feature trees to obtain qualified combinations of feature points and then search for their relevant objects in the object tree. Based on our experimental findings, BB* is scalable to large data sets and it is the most robust algorithm with respect to various parameters. However, FJ is the best algorithm in cases where the number m of feature data sets is low and each feature data set is small.

REFERENCES

1. M.L. Yiu, X. Dai, N. Mamoulis, and M. Vaitis, “Top-k Spatial Preference Queries,” Proc. IEEE Int’l Conf. Data Eng. (ICDE),2007.
2. N. Bruno, L. Gravano, and A. Marian, “Evaluating Top-k Queries over Web-Accessible Databases,” Proc. IEEE Int’l Conf. Data Eng. (ICDE), 2002.
3. Guttman, “R-Trees: A Dynamic Index Structure for Spatial Searching,” Proc. ACM SIGMOD, 1984.
   [4] G.R. Hjaltason and H. Samet, “Distance Browsing in Spatial Databases,” ACM Trans. Database Systems, vol. 24, no. 2, pp. 265-318, 1999.
4. R. Weber, H.-J. Schek, and S. Blott, “A Quantitative Analysis and Performance Study for Similarity-Search Methods in High- Dimensional Spaces,” Proc. Int’l Conf. Very Large Data Bases (VLDB), 1998.
5. K.S. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft, “When is ‘Nearest Neighbor’ Meaningful?” Proc. Seventh Int’l Conf. Database Theory (ICDT), 1999.

6. R. Fagin, A. Lotem, and M. Naor, “Optimal Aggregation Algorithms for Middleware,” Proc. Int’l Symp. Principles of Database Systems (PODS), 2001.
7. I.F. Ilyas, W.G. Aref, and A. Elmagarmid, “Supporting Top-k Join Queries in Relational Databases,” Proc. 29th Int’l Conf. Very Large Data Bases (VLDB), 2003.
8. N. Mamoulis, M.L. Yiu, K.H. Cheng, and D.W. Cheung, “Efficient Top-k Aggregation of Ranked Inputs,” ACM Trans. Database Systems, vol. 32, no. 3, p. 19, 2007.
9. D. Papadias, P. Kalnis, J. Zhang, and Y. Tao, “Efficient OLAP Operations in Spatial Data Warehouses,” Proc. Int’l Symp. Spatial and Temporal Databases (SSTD), 2001.
10. S. Hong, B. Moon, and S. Lee, “Efficient Execution of Range Top-kQueries in Aggregate R-Trees,” IEICE Trans. Information and Systems, vol. 88-D, no. 11, pp. 2544-2554, 2005.
11. T. Xia, D. Zhang, E. Kanoulas, and Y. Du, “On Computing Top-k Most Influential Spatial Sites,” Proc. 31st Int’l Conf. Very Large DataBases (VLDB), 2005.
12. Y. Du, D. Zhang, and T. Xia, “The Optimal-Location Query,” Proc. Int’l Symp. Spatial and Temporal Databases (SSTD), 2005.
13. D. Zhang, Y. Du, T. Xia, and Y. Tao, “Progessive Computation of The Min-Dist Optimal-Location Query,” Proc. 32nd Int’l Conf. Very Large Data Bases (VLDB), 2006.
14. Y. Chen and J.M. Patel, “Efficient Evaluation of All-Nearest-Neighbor Queries,” Proc. IEEE Int’l Conf. Data Eng. (ICDE), 2007.
15. N. Mamoulis and D. Papadias, “Multiway Spatial Joins,” ACM Trans. Database Systems, vol. 26, no. 4, pp. 424-475, 2001.