Research on Group Reverse Farthest Neighbour Query Algorithm

Yongshan Liu¹, Xiang Gong¹*, Dehan Kong², Tianbao Hao¹ and Xiaqi Yan¹

¹Department of Information Science and Engineering, Yanshan University, Qinhuangdao, China
²Department of Information Engineering, Hebei University of Environmental Engineering, Qinhuangdao, China

*Corresponding author email: ysugongx@163.com

Abstract. In recent years, the demand for location-based services is growing. From resource tracking to personal life assistance, spatial data query technology plays an important role, especially the application of reverse nearest neighbor query technology in decision support, resource allocation optimization, data mining. By contrast, the reverse farthest neighbor query has become a more and more hot topic in the research of spatial database theory in recent years. The purpose of the reverse farthest neighbor query is to obtain the final result sets with the given point as its farthest neighbor, which is specially used to solve the problem of weak influence set in space. This paper gives a query optimization algorithm based on the least covering circle. The algorithm firstly introduces the minimal covering circle algorithm. Secondly, the p-ray pruning algorithm is used to filter the candidate set in the second level, and the candidate set is modified by the reverse far-neighborhood range query algorithm to obtain the final result set, which solves the problem in Euclidean effectively. Through specific comparative experimental analysis, c-grfn algorithm has good query performance not only in random distributed environment, but also in Gaussian distribution. It usually has 20% advantages of time in distributed space. As the data set increases, the time advantage becomes more obvious.

1. Introduction
As location based services of all kinds of mobile terminal software, with the rapid development of the spatial database query technology also has been widely used, mainly in the calculation based on location, geography, urban planning, and other fields, and it on some challenging research problems also play an increasingly important role, has a very broad application prospects and potential value. Among them, the nearest neighbor query, the reverse nearest neighbor query, the reverse farthest neighbor query and its various variants are common spatial query types, which have extensive application needs in market decision-making, facility location, and ecological research.

The problem of the reverse farthest neighbor query, which aims to discover the weak influence set, has been studied in depth in recent years. As a relatively new class of problems in spatial database, weak impact set has important application value in urban planning, like the placement of supplies after earthquake, allocation of resource and other fields. For example, there is always a problem which concerns the location picking of the chemical plant, because of the toxic substances. Therefore, it is very necessary to relocate the facility as far away from residential area as possible. Another example, in the tourism industry, tourists like the shopping in their travel, however, they usually choose the place nearest to them, except for some other reasons. As a store operator, who don’t want as many
tourists as possible to visit their business. Therefore, he should install monitoring equipment to obtain the farthest neighbor of his opposite, and increase publicity in these places, or add shuttle bus. Consider BotFighters, a mobile game designed to play in location-based environments. The task of the game is to find and then shoot/kill other players. Players always tend to ignore other players who are farthest away from them, and those who are ignored can be deadly to them. Users can take advantage of this by issuing an RkFN query to find the player who sees him as one of the furthest objects to shoot, which might be even better. The same strategy can be applied to real-world battlefields. It can be seen that the effective solution to the proposed problems, has important application value. The existing methods only deal with a single point, and does not take the many query points as a group into consideration. However, in real world, there are also some problems of reverse nearest neighbor query based on multiple query points, the research group reverse farthest neighbor query has practical significance.

This paper defines group reverse farthest neighbor query firstly. In order to solve this problem more effectively and improve the query efficiency, the query point set is approximated as a query point by obtaining its minimum coverage circle, and the group query problem is skillfully transformed into the reverse farthest neighbor problem of a query point. Then through the framework of filtering and purification, a pruning algorithm based on quartering region and p-ray is proposed to reduce the data set. Finally, through the proposed refining algorithm, the candidate set can be purified. The purpose of this article is obtained which is to get the results effectively.

The structure of the paper is as follows: Part 2 introduce the research status. Part 3 is about some basic definitions and theorems, as concepts and theorems of related data query are briefly introduced. Part 4 introduces the processing process of the group reverse farthest neighbor query algorithm based on filtering and purifying framework and describes the algorithm analysis of pruning and refining process in detail. In final part, experimental test is carried out with experimental simulation data, which proves that the query algorithm has good query performance.

2. Related Work

In 2000, Korn, P and Muthukrishnan, S first formalized a novel notion of influence based on reverse neighbor queries and its variants[1]. Boren Li proposed a novel incremental RkNN algorithm, applied to multidimensional spatial databases[2]. Otherwise, Bin Yao presented the challenges associated with such queries and proposes efficient, R-tree based algorithms for both monochromatic and bichromatic versions of the Rrfn queries[3]. Quoc Thai Tran aimed to introduce a new approach to process reverse proximity queries including RFN and RkNN/RkFN queries[4]. Their approach is based on NVD and pre-computation of network distances, and is applicable for spatial road network maps. In 2010, Liu proposed an efficient algorithm for RFN query with metric index, which used the properties of convex hull and triangle inequality to trim, shortening the query time effectively [5]. In 2011, Gao studied a new type of spatial query, namely aggregate k farthest neighbor (AkFN) search. Given a data point set P, a query point set Q, an AkFN query returns k points in P with the largest aggregate distances to all points in Q. He defined by the aggregation functions SUM, MAX and MIN, and presented the MB and BF algorithms based on R tree[6]. In 2012, Liu J proposed the reverse farthest neighbor query algorithm without location restrictions, and compared it with the PIV algorithm proposed in 2010[7]. In 2013, Alan presented an inverted neighborhood model, k-Furthest Neighbors to identify less ordinary neighborhoods for the purpose of creating more diverse recommendations[8]. Using a collaborative filtering scheduling algorithm based on k distant neighborhood. In 2014, Wang addressed the problem of searching the k aggregate farthest neighbours on road networks[9]. Given a query point set, aimed at finding the top- k points from a dataset with the largest aggregate network distance. He discussed the problem of searching k furthest neighbors on the road network. In 2016, Li, B proposed dynamic reverse farthest neighbor query algorithm and some related algorithms by pruning query space precisely based on filtering and purification framework algorithm[10]. In the same year of 2016, Wang S presented an efficient algorithm to process the RkFN queries based on several interesting observations[11]. He also presented a rigorous theoretical analysis to study various important aspects of the problem and their algorithm first studied the reverse farthest neighbor query problem based on any k value. In 2017, Xu, X.J. proposed an algorithm to solve the monochrome and two-color reverse
farthest neighbor query in the road network by using landmark and zoning technology\cite{12}. Shengsheng Wang proposed an algorithm called PRCLU for PRkNN with larger k, including pruning phase and verification phase\cite{13}. The pruning phase with a minimum circle to enclose the uncertain data, which performs pruning with the region, then followed by probabilistic pruning strategy in sequence. In the same year, Haryanto proposed Group Reverse kNN as a solution, it gave the farthest neighbor query problem based on MOIS-tree index structure, and the definition of minimum maximum distance and maximum distance are given\cite{14}. Li propose efficient R_k NN query verification techniques which utilize the influence zone to check the integrity of query results. The methods in this work aim to verify both monochromatic and bichromatic R_k NN queries results\cite{15}.

In 2004, Dimitris Papadias presented three algorithms for solving group nearest neighbor query, namely MQM, SPM and MBM\cite{16}. In 2005, Li H employed an ellipse to approximate the extent of multiple query points, and then derive a distance or minimum bounding rectangle (MBR) using that ellipse to prune intermediate nodes in a depth-first search\cite{17}. The GNN method of pruning with the ellipse formed by the farthest point in the set of query points and its MBR as the breakthrough point. In 2005, Papadias D proposed an aggregated nearest neighbor query algorithm based on a set of query points\cite{18,19}. In 2008, Lian X focused on another important query, namely probabilistic group nearest neighbor query (PGNN), in the uncertain database, which also has many applications\cite{20}. A probabilistic nearest neighbor query algorithm for uncertain data was proposed. In 2010, Song xiaoyu proposed the minimum coverage circle to reach the perspectives of collinear and incollinear of the query point set\cite{21}. In 2011, a method was proposed to obtain the optimal solution of the nearest group query of the constraint group by iteratively updating the cluster\cite{22} by Mo Chen. In 2013, Zexue Yang proposed an algorithm to solve the obstacle space’s nearest neighbor query based on the different location relations\cite{24}. Zhang liping in order to reduce complexity and improve efficiency, based on the divide and conquer, local optimization and dynamic update of scan lines\cite{25}, the method to construct Voronoi diagram using the convex hulls was proposed in 2017. In the same year, a probabilistic threshold group k-nearest neighbor query method( PTGk NN method) is proposed which is based on uncertain Voronoi diagram\cite{26}. A processing framework and some optimal techniques including pruning and user preference grouping methods are presented by M. Li\cite{27}.

To sum up, the algorithms mentioned above are just about a single query point, but for group queries which are all about nearest neighbor and its variants, they are helpless to solve the reverse farthest neighbor query problem for a group of points. Therefore, combining with the previous excellent research results, this article proposes the reverse farthest neighbor query firstly and then conducts in-depth research.

3. Definitions and Theorems

3.1. Problem Definition

Study the contents of this paper is based on the European space are discussed, the concept of group is to query extends to reverse the farthest in the query, based on the European space group of the concept of reverse furthest neighbor query, in order to presents the definition of the problem more clearly, this section draw out the central thesis of this chapter. Before the definition is given, it needs to be explained that since the research is carried out by Euclidean space, the distance of any two points p and q is called the Euclidean’s distance, which can be written as dist(p, q), as shown in formula (1).

\[
\text{dist}(p, q) = \sqrt{(p_x - q_x)^2 + (p_y - q_y)^2}
\]

Group Nearest Neighbor query(GNN) is a variant of Nearest query, it is Nearest neighbor query for multiple objects and it is used to query Nearest Nearest neighbor Nearest Nearest neighbor to Nearest Nearest Nearest neighbor. As described in definition 1.
Definition 1 GNN. Given a data set \( P = \{ p_1, p_2, \ldots, p_n \} \), a set of query points \( Q = \{ q_1, q_2, \ldots, q_m \} \), the total distance between data point \( p_i \) in \( P \) and all points in query point set \( Q \) is defined as 
\[
  d_e(p, Q) = \sum_{r=1}^{n} d_e(p, q_r) \]
GNN query is to find a subset satisfying formula (2).

\[
  GNN(Q) = \{ p \in P | \forall q_r \in P - \{ p \} : d_e(p, Q) \leq d_e(p, Q) \}
\]  
(2)

The concept of Group \( k \) Nearest Neighbors query can also be extended to Group \( k \) Nearest Neighbors (GkNN) query \((k>1)\), that is, to find \( k \) data objects in set \( P \) that meet formula (2), that is, to query \( k \) query objects with relatively small sum of distance from the data point set.

Group Reverse Nearest Neighbors (GRNN) query is an extension of Group query in Reverse Nearest Neighbors query, which is used to obtain the final result with any point in the query point set as the farthest neighbor. It can be used to evaluate the influence of a group of target objects or obtain the strong influence set of a Group of target objects in practical application. As described in definition 2.

Definition 2 GRNN. Given a set of data points \( P = \{ p_1, p_2, \ldots, p_n \} \) and a set of query points \( Q = \{ q_1, q_2, \ldots, q_m \} \), if any point in the query point set is in the nearest neighbor result set of the data point \( p_i \), then the point \( p_i \) is a result of \( GRNN(P, Q) \). As shown in formula (3).

\[
  GRNN(Q) = \{ p | \exists q \in Q, \exists p \in P | q \in NN(p) \}
\]
(3)

The concept of Group Reverse Nearest Neighbors can also be extended to Group Reverse \( k \) Nearest Neighbors (GRkNN) query \((k>1)\), that is, to find \( k \) data objects in set \( P \) that meet the requirements of formula (3), as shown in formula (4).

\[
  GRkNN(Q, k) = \{ p | \exists q \in Q, \exists p \in P | q \in kNN(p, k, P) \}
\]
(4)

Farthest Neighbors (FN) query is the concept opposite to the nearest neighbor query, that is, the furthest point that is found in the data set from the query point. As described in definition 3.

Definition 3 Farthest Neighbor (FN) Query. Assuming that given a data point set \( P \) and a query point \( q \), the farthest Neighbor (FN) query is the subset of a set of \( P \) that satisfies formula (5).

\[
  FN(q) = \{ r \in P | \forall p \in P, \text{dist}(r, q) \geq \text{dist}(p, q) \}
\]
(5)

The concept of farthest neighbor query can be extended to \( k \) farthest Neighbors (kFN) query \((k>1)\). kFN query is to find \( k \) data points satisfying formula (5) in the data set, that is, to get the most distant object of \( k \) query points. Reverse Farthest Neighbors (RFN) query is the concept opposite to reverse nearest neighbor query. In the actual situation, the target point was find by using the weak influence. It is described in definition 4.

Definition 4 Reverse Farthest Neighbors (RFN) Query. Given a data set \( P \) and a query point \( q \), RFN query is to find a subset \( RFN(q) \) of a set of \( P \) that satisfies formula (6).

\[
  RFN(q) = \{ p \in P | q \in FN(p) \}
\]
(6)

Of course, the concept of reverse farthest neighbor query can also be extended to the question of inverse \( k \) farthestmost Neighbors (RkFN) query, that is, the query point in the data set is put into the point set of \( k \)-farthestmost neighbor result set, as shown in formula (7).

\[
  RkFN(q) = \{ p \in P | q \in kFN(p) \}
\]
(7)

In general, there is no necessary connection between the farthest neighbor and the farthest opposite neighbor, that is, if point \( q \) is the farthest neighbor of point \( p \), but it cannot be deduced that point \( p \) must be the farthest opposite neighbor of point \( q \), and vice versa.

Definition 5 Group Reverse Farthest Neighbors (GRFN) Query. Suppose a set of query objects \( Q = \{ q_1, q_2, \ldots, q_n \} \) and a set of data points \( P = \{ p_1, p_2, \ldots, p_m \} \), the group reverse farthest neighbor query is to find the spatial data points in the farthest neighbor containing any query point \( Q \) in \( Q \) in the given data set \( P \). As shown in formula (8).
As shown in Figure 1, in a given two-dimensional space, there is a set of query points $Q = \{q_1, q_2, q_3\}$, and a set of data points $P = \{p_1, p_2, p_3\}$. Take point $p_1$ as an example, $\text{dist}(p_1, p_3) < \text{dist}(p_1, q_3) < \text{dist}(p_1, p_2) < \text{dist}(p_1, q_2) < \text{dist}(p_1, q_1)$. That is, the point which is farthest away from the point $p_1$ is the point $q_1$. Since $q_1$ is in the query set, point $p_1$ is one of the final candidates of the set $Q$.

**Figure 1.** Reverse nearest neighbor query.

The minimum coverage circle algorithm for obtaining a set of points effectively plays an increasingly important role in spatial database, location of distribution network substation and other problems. Since the center point is the point with the maximum distance of all points in the set of distance points, the group query problem can be effectively transformed into a point query problem by obtaining the center of the minimum covering circle of a set of query points and taking this point as a new query point. Although it is an approximate estimate, it can still better meet the application needs in practical applications. The concept of it is as follows.

**Definition 6 Minimum Coverage Circle.** Given a set of query objects $Q = \{q_1, q_2, ..., q_n\}$, a circle which contains all objects in $Q$ with a minimum radius is defined as the minimum covering circle, denoted as $\text{Cir}(O, r)$. Point $O$ is the center of the circle and $r$ is the radius, which can say satisfying formula (9).

$$\text{Cir}(q, r) = \{q | \text{dist}(q, O) \leq r, q \in Q\}$$  \hspace{1cm} (9)

As figure 2’s shows, set $P = \{p_1, p_3\}$, and the circle which take point $O$ as a center is the minimum coverage circle of $P$.

**Definition 7 Quarter Neighborhood** In the two-dimensional space, two perpendicular line segments Line1 and Line2 are made to consider $q$ as the center, and these two line segments are defined as the boundary line of the quartile neighborhood. The space is divided into four areas by the boundary line, as is shown in Figure 3: A, B, C and D. These four areas are called the quadric neighborhood of point $q$. Two of the four areas are diagonal to each other: $A$ and $C$ are diagonal to each other, and $B$ and $D$ are diagonal to each other. Similarly, the space can also be divided into eight neighborhood areas, sixteen neighborhood areas and so on according to the demand.

**Figure 3.** Quarter neighborhood.

**Figure 4.** Schematic diagram of theorem 1.
3.2. Related Theorem

Sections should be numbered with a dot following the number and then separated by a single space:

Theorem 1. In the four quadrants divided by the query point as the origin, if there are data points in both quadrants of the diagonal, then the points in these two quadrants must not be the farthest opposite neighboring points of the query point.

Prove: As shown in figure 4, based on the origin of the query point \( q \) in the four quadrants, the first quadrant and the third quadrant are diagonal area, and several points, points \( p_1, p_2, \) for example, because the \( \text{dist}(p_1, p_2) > \text{dist}(p_1, q) \), and \( \text{dist}(p_1, p_2) > \text{dist}(p_2, q) \), so the points \( p_1 \) and \( p_2 \) is the longest adjacent won't point \( q \), by the same token, if the diagonal quadrant in several positions to exist, the arbitrary point of the two quadrants is not query point \( q \) reverse furthest neighbor query result set. Theorem 1 is proved.

Theorem 2. In two-dimensional space, suppose point \( p \) is any point in the data set, make a ray \( L_{qp} \) passing through the data point \( p \) through the query point \( q \), and make a line segment \( L_q \) passing point \( q \) and perpendicular to the ray, which divides the space into two half planes. The plane where point \( p \) is located is defined as \( H_p \) plane, and the other plane is defined as \( H_q \) plane. If point set exists in \( H_q \) plane, then point \( p \) needs to be pruned.

Proof: as shown in figure 5, as shown in theorem 2, point \( f_1 \) is a point in the \( H_q \) plane, point \( p \) is a point on ray \( L_{qp} \), so point \( f_1 \) can trim off point \( p \). As shown in figure 5, point \( x \) is the intersection of line \( pf_1 \) and line \( L_q \). Since ray \( L_{pq} \) and line \( L_q \) are perpendicular to each other, \( \triangle xqp \) is a right triangle, or point \( x \) coincides with point \( q \), so \( \text{dist}(p, x) \geq \text{dist}(p, q) \), so \( \text{dist}(p, f_1) > \text{dist}(p, q) \) holds. Theorem 2 is proved.

\[ \begin{align*}
\text{Figure 5. Proof diagram of theorem 2.} & \quad \text{Figure 6. Proof diagram of theorem 3.}
\end{align*} \]

In this paper, the Range-k algorithm is improved and a verification algorithm suitable for the reverse farthest neighbor query is proposed. Because the reverse farthest neighbor query is an operation that takes the query point \( q \) as the FN data point set in \( P \), the farthest neighbor of any point in the result set must be the query point \( q \). According to this idea, theorem 3 is given.

Theorem 3. In two-dimensional space, given a \( P \) which means a set of data point and a query point \( q \), if the point \( p_i \) in the data set is a result set of RFN(\( q \)), then other points in the data set must be in the circle with point \( p_i \) as the center and the distance between point \( p_i \) and point \( q \) as the radius, otherwise, \( p_i \notin \text{RFN}(q) \).

As shown in figure 6, given a set of data points \( P = \{p_1, p_2, p_3, \ldots, p_{14}\} \), and query point \( q \). To query the reverse farthest neighbor of \( q \), according to the definition of the reverse farthest neighbor query, for any point \( P \) in the data set \( P \), if the farthest neighbor of \( P \) contains \( q \), then \( P \) is the reverse farthest neighbor of \( q \). In figure 6, \( p_i \) is taken as the center and the distance between \( q \) and \( p_i \), \( \text{dist}(p_i, q) \), is concerned as the radius. Theorem 3 is proved.

4. The Relevant Algorithm

In this chapter, the definition of group reverse farthest neighbor query in Euclidean space is proposed for the first time. In order to solve this problem effectively, this section proposes a group reverse furthest neighbor query algorithm based on minimum coverage circle: C-GRFN algorithm. Firstly, using minimum covering circle algorithm to obtain the center of query points. Secondly, the result set is judged by the pruning strategy based on the quarter neighborhood in theorem 1. If it is judged to be
empty, the empty result set will be returned and the following query will be stopped. Based on the p-ray pruning method proposed, the number of data set is reduced by second-order filtering. Finally, to gain the candidate set by the refinement algorithm based on theorem 3, and the final query result is obtained. The specific C-GRFN algorithm is shown in table 1.

Table 1. C-GRFN algorithm

| Input: Querysets Q; Datasets P | Output: Result ResultPoints |
|-------------------------------|-----------------------------|
| BEGIN                         |                             |
| 1: q←GetCircle(Q);           |                             |
| 2: Candidate←CutByRegion(P); |                             |
| 3: If(Candidate is null) Then |                             |
| 4: return null;              |                             |
| 5: Else                      |                             |
| 6: Candidate←CutByPray(Candidate) |                     |
| 7: If(Candidate is null) Then |                             |
| 8: return null;              |                             |
| 9: Else                      |                             |
| 10: ResultPoints←F-Range(Candidate); |             |
| 11: Return ResultPoints;     |                             |
| END                           |                             |

First, the center of the query point set Q is obtained (line 1). To the query point as the center, divided into four quadrants, according to the data points in four quadrants judging the distribution of the number of query results (line 1, 2-4), if not null, then again on the basis of P-ray algorithm are pruned filtering data set (5-6), if the candidate sets empty after filtering, the result sets (line 7-8), otherwise the candidate point set refined to obtain the final result set (line 10-11).

In the following paragraphs, the algorithms involved in C-GRFN query algorithm and their algorithm analysis will be presented successively, including the algorithm of minimum coverage circle, quadrantal neighborhood pruning algorithm, p-ray pruning algorithm and the detailed description of Frange-k refining algorithm.

4.1. The Algorithm of Minimum Coverage Circle

The minimum covering circle algorithm is used to calculate the center of the query point set. It can fit the requirements in real life. This algorithm can transform the group reverse farthest neighbor query into a reverse farthest neighbor query of a query point effectively. The algorithm is shown as follows.

Table 2. Algorithm GetCircle(Q)

| Input: Querysets Q; | Output: Center of the circle q; |
|--------------------|---------------------------------|
| BEGIN              |                                 |
| 1: For i←0 to querySets.count Do |                         |
| 2: If incircle(querySets[i])=false Then |                   |
| 3: P1←querySets[i]; |                                   |
| 4: For j←0 to i Do |                                       |
| 5: If incircle(querySets[j])=false Then |                   |
| 6: P2←getCenterPoint(querySets[i],querySets[j]); |           |
| 7: Radius←getDist(P1,querySets[i]); |                                |
| 8: For k←0 to j Do |                                         |
| 9: If incircle(querySets[k])=false Then |                             |
| 10: centerPoint←getCenterPoint() |                           |
| 11: Radius←getDist(P3,querySets[i]); |                      |
| 12: Return q |                                         |
| END               |                                 |
Traditional method of calculating minimum covering circle is $O[\lg(d/R)n]$, $d$ represents the distance, and $R$ represents the radius. The time complexity of our algorithm for calculating the minimum coverage circle is $O(n)$. The steps can be described as follows:

1. Firstly, $Q$ represents the query point set which is randomly distributed is obtained.
2. Secondly, the data points are added to the circle in turn, and the position of the point and the circle needs to be calculated for each point added.
3. Thirdly, if $p_i$ is outside the circle, which means $p_i$ must on the boundary of the minimum covering circle, then we go to step (4), otherwise no update is needed, and return (2).
4. This moment, $p_i$ must on the boundary of the current minimum covering circle. At this point, $p_i$ can be the center of the new circle with a radius of zero. Then, the previous $i$-1 point is added to the circle one by one through the above steps.
5. If $p_j$ is outside the current minimum circle, which means $p_j$ must on the boundary of the current minimum covering circle, then we go to step (6), otherwise no update is needed, and return step (4).
6. Point $p_i$, $p_j$ must be on the border, at this time it will be the first point with the first $j$ point of attachment point set as the new circle's center, the distance between two set then through the above steps will $j$-1 point before to join in this circle, each join point need to perform the first step (7).
7. If $p_k$ is on the circle, which means $p_k$ must on the boundary of the minimum covering circle of the first $k$ points, cause three points can determine a circle.

4.2. Quadrature Neighborhood Pruning Algorithm

According to the method of pruning data set based on the distribution of data points in the diagonal quadrant mentioned in theorem 3, a pruning algorithm based on the quartile neighborhood region, CutByRegion algorithm, is proposed. This algorithm can judge whether the result set of the query is empty or not, which effectively avoids the time consumption caused by direct query without judgment and the final calculation is fruitless. The specific algorithm is shown in table 3.

| Table 3. Algorithm CutByRegion($P$) |
|-------------------------------------|
| **Input:** Datasets $P$;            |
| **Output:** candidatesets: Candidate|
| **BEGIN**                           |
| 1. Candidate←$P$;                   |
| 2. region1←getRregion1($P$);        |
| 3. region2←getRregion2($P$);        |
| 4. region3←getRregion3($P$);        |
| 5. region4←getRregion4($P$);        |
| 6. If region1!=null && region3!=null Then |
| 7. Candidate←prunePoints(Candidate,region1,region3); |
| 8. If region2!=null && region4!=null Then |
| 9. Candidate←prunePoints(Candidate,region2,region4); |
| 10. pOnLine1←getRightLine(Candidate); |
| 11. pOnLine2←getUpLine(Candidate);   |
| 12. pOnLine3←getLeftLine(Candidate); |
| 13. pOnLine4←getDownLine(Candidate); |
| 14. If pOnLine1!=null && pOnLine3!=null Then |
| 15. Candidate←prunePoints(Candidate, pOnLine1, pOnLine3); |
| 16. If pOnLine2!=null && pOnLine4!=null Then |
| 17. Candidate←prunePoints(Candidate, pOnLine2, pOnLine4); |
| 18. If Candidate=null Then |
| 19. Return null;                   |
| 20. else                            |
| 21. Return Candidate;              |
The time cost of query in order to avoid costly to obtain zero result set, need to be at the beginning of the query data points set \( P \) use four primary filtering algorithm CutByRegion neighborhood area: first, in order to query point set minimum cover circle in the center of origin divided four quadrants, respectively to obtain data points of the four quadrant (line 1, 2-5), according to the distribution of data set is empty in the diagonal quadrant and pruning data set (line 6-9). Second, the candidate point set to iterate over access to the candidate of centralization in point on the axis, and respectively in four sets (line 10-13), according to two half axis parallel to each other on the distribution of point set to filter candidate chi-chi (14-17 rows), if the filtered result sets empty, is to stop the query, otherwise will filter the results for the candidate set (line 18-21).

### 4.3. \( P \)-ray Pruning Algorithm

#### Table 4. Algorithm CutByPray (\( P \))

| Input: Candidate: regionCandidate; center: \( q \) |
| Output: PCandidateSets |

BEGIN
1: PCandidateSets ← regionCandidate;
2: For each \( p \) in PCandidateSets Do
3: \( k_0 \leftarrow \) getk(\( p, q \));
4: \( k_1 \leftarrow -1/k_0 \);
5: \( b \leftarrow \) getbOfPray(\( k_1, q \));
6: For each \( t \) in \( P \) Do
7: flag1 ← UpOfLine(\( t \));
8: flag2 ← UpOfLine(\( p \));
9: If (flag1!=flag2)
10: PCandidateSets.remove(\( p \));
11 break;
12: Return PCandidateSets;
End

Using CutByPray algorithm iterate over the data set point (line 2) are pruned: first point \( p \) and the center of the circle \( q \) two attachment for slope value \( k_0 \) (line 3), and then get the attachment of the vertical slope values \( k_1 \) and intercept \( b \) (4-5), the space is divided into two and a half by the vertical plane, then iterate over the data set point at this time, if the point \( t \) and \( p \) is not in a plane, then on the basis of theorem 3.4 point \( p \) can be pruned, jump out of the inner loop, perform the outer loop of the next data point line (6-11). And so on to get the final candidate set (line 12).

### 4.4. Refining Algorithm

Wu and others in the literature [32] is proposed for refining k nearest neighbor query result set Range - k algorithm, this algorithm can be described as: thought, centered on candidate focused point to that point and the distance from the query point for the radius of a circle, if the data set points in the circle to be less than \( k \), then that point can be incorporated into the results, otherwise, need to be revised. Theorem 3 is based on this algorithm to improve the proposed. Based on theorem 3, this section proposes a verification algorithm for reverse farthest neighbor: f-range algorithm. Using the algorithm to filter the set of candidate for refining, refining algorithm thinking can be described as: test for each candidate, the distance of the candidate point to point \( q \) is the radius, with the candidate points for the center of the circle circle, if the number of data points outside the non-zero, as far as the candidate not \( q \), in other words, the point \( q \) the reverse is not the candidate, as far as the point need to be removed. The specific refining algorithm is shown in table 5.
Table 5. Algorithm F-Range(PCandidateSets)

| Input: candidate: PcandidateSets; center: q |
|-------------------------------------------|
| Output: result: resultPoints              |

BEGIN
1: For each \( p \) in CandidateSets Do
2:     radius ← dist\((p, q)\);
3:     If getPNum_OutCircle\((p, radius)\)>0 Then
4:        CandidateSets.remove\((p)\);
5:     resultPoints ← Candidate;
6:   Return resultPoints;
END

To iterate over the candidate focus point (line 1), radius is candidate point to the query point distance (line 2), as a circle, centered on candidate statistical outside circle the number of point set, if the outside circle point set is not empty, the query point is not the candidate furthest neighbor, namely the candidate is not reverse furthest neighbor query point, need to be removed from a candidate set (line 3-4). The candidate set at the end of the loop is the result set (line 5).

5. Analysis of Experimental Results

5.1. Experimental Environment Parameter Configuration

The experiment environment carried out by this article is as follows: Microsoft Windows7 operating system. Using C# computer language and Visual Studio2015 environment. In this paper, SpatialDataGenerator software is used to generate random two-dimensional space objects, which are distributed in the regions of 0<x<700, 0<y<700.

5.2. Analysis of Experimental Results

In this paper, the method of basic-group inverse farthest neighbors (B-GRFN) is regarded as a comparison algorithm, and using the C-GRFN algorithm proposed by this article to make a comparison. The algorithm idea of the traditional algorithm can be explained as: firstly, the group query problem is transformed into a point query problem by obtaining the minimum coverage center of the query point set; secondly, the result set of the query is obtained by using the pruning algorithm based on convex hull. Comparative experimental data and analysis are described below. First, the impact of query point set size on query should be tested efficiency. In order to gain the principle of single variable, which is: set \( P=1 \times 10^4 \), and presented random distribution. As shown in figure 7, with the increase of the number of query point, two algorithms’ time increase slowly, which indicates the change of the query point set has a less impact on the running time of the algorithm. Through the comparison, it can be knows that the query time of C-GRFN algorithm is far less than that of B-GRFN algorithm, so the query performance of CGRFN algorithm is obviously better than that of B-GRFN algorithm.

Figure 7. Impact of query point set size

Figure 8. Impact of random distributed data
Secondly, the effect of data point set size under random distribution on query efficiency was tested. In order to maintain the principle of single variable, the experiment set other attribute values of the two groups of algorithms to be consistent, namely: query point set $Q=1\times10^3$, and presented random distribution. As shown in figure 8, the query time of both algorithms increases with the increase of data point set. C-GRFN algorithm mainly increases the execution time of pruning algorithm due to the increase of data set capacity. The main reason of B-GRFN algorithm is that the increase of data set capacity will make the time to construct convex hull longer. The comparison shows that the query time of C-GRFN algorithm is far less than that of B-GRFN algorithm, and the time growth rate of C-GRFN algorithm is relatively slow. Therefore, the C-GRFN algorithm proposed in this paper has better query performance.

Thirdly, the influence of the size of data point set under gaussian distribution on query efficiency is tested. In order to maintain the principle of single variable, the experiment set other attribute values of the two groups of algorithms to be consistent, namely: query point set $Q=1\times10^3$, and keep random distribution, which is consistent with figure 8. As shown in figure 9, with the increase of data set capacity, the time to build convex hull gradually increases, so the query time of B-GRFN algorithm is also increasing, and the change is significant. In contrast, the running time of C-GRFN algorithm does not change significantly due to the increase of data point set capacity. Therefore, the C-GRFN algorithm proposed in this paper has better query performance.

Then, test the impact of data point sets with different sizes and data distributions on query efficiency in C-GRFN algorithm. In order to maintain the principle of single variable, set $Q=1\times10^3$, and presented random distribution. In the experiment, the data set gradually increased from 10,000 to 30,000, and the two sets of data sets presented gaussian distribution and random distribution, respectively. As shown in figure 10, the query time under the gaussian distribution is slightly less than that under the random distribution, mainly because when the data distribution is relatively concentrated, the pruning algorithm in the quarter neighborhood region can effectively filter a large number of data point sets, but the query efficiency of the two is comparable. Looking at figure 8 and 9, no matter in the data set of random distribution or the data set of gaussian distribution, C-GRFN algorithm has obvious query advantages, which indicates that C-GRFN has good robustness.

Finally, the effect of gaussian variance on query efficiency is tested. There are two parameters in the gaussian distribution: mean and variance. Since mean affects the absolute position of data distribution, it will not have too obvious influence on the experimental results. Therefore, only the influence of variance on the efficiency is tested here. Variance determines the density of data distribution. The smaller the value, the denser the distribution. In order to maintain the principle of single variable, set $Q=1\times10^3$, and maintain random distribution. Data set $P=1\times10^4$, presenting gaussian distribution. As figure 11 shows, with the gradual increase of variance, the query time of both algorithms increases, but the query time of C-GRFN algorithm is far less than that of B-GRFN algorithm, and the time
change is relatively stable, indicating that C-GRFN algorithm has better query performance and good robustness.

![Figure 11](image.png)

**Figure 11.** Influence of the variance of gaussian distribution.

### 6. Conclusion

The definition of group reverse farthest neighbor query is proposed and the query optimization algorithm based on the minimum covering circle is proposed: C-GRFN algorithm. Firstly, the minimum coverage circle algorithm is used to obtain the center of the minimum coverage circle of the query point set. Secondly, the distribution of data point sets in the diagonal region is used to determine whether the result set is empty. If it is empty, the query will be stopped; otherwise, the data set will be filtered. Then the candidate set is filtered by p-ray pruning strategy. At last, the candidate set is refined by the improved F-range anti-far-range query algorithm based on range-k algorithm to obtain the final result set. The experimental results show that the algorithm has good query performance, can effectively solve the problem of group reverse farthest neighbor queries in Euclidean space, and fills the technical gap in the field of reverse farthest neighbor queries. The research work of this paper is summarized as follows:

1. The pruning algorithms of base quartering neighborhood and p-ray are given, and the candidate sets are obtained by filtering the data point sets with these two pruning strategies. In the refining stage, the range-k algorithm was studied and improved, and the F-range algorithm was proposed to modify and purify candidate sets, so as to ensure the accuracy of query result sets.
2. Then this paper establish a filtration-purification framework. Data sets of different sizes, query point sets, and data distribution under different conditions are analyzed to verify that the proposed algorithm in this paper has good query performance.

Although it is helpful in two-dimensional space, there are still some limitations, for example, all the spatial data objects are based on European space, while in reality there is no lack of road network and the presence of obstacles, in order to solve this problem can be discussed in future studies.

### Acknowledgments

This work was supported by the National Natural Science Foundation under Grant 61972334 and Natural Science Foundation of Hebei Province under Grant F2017203019 and 2019 Hebei University Higher Education Science and Technology Research Youth Fund Project under Grant No.QN2019044.

### References

[1] Korn, F., & Muthukrishnan, S. (2000). Influence sets based on reverse nearest neighbor queries. ACM SIGMOD Record.

[2] Li, B., Pan, M., & Wu, Z. (2011). Effective reverse k-nearest neighbor query based on revised r*-tree in spatial databases. International Conference on Geoinformatics, 1-5.

[3] Yao, B., Li, F., & Kumar, P. (2009). Reverse Furthest Neighbors in Spatial Databases. IEEE International Conference on Data Engineering. IEEE.
[4] Tran, Q. T., Taniar, D., & Safar, M. (2009). Reverse k nearest neighbor and reverse farthest neighbor search on spatial networks.

[5] Liu, J., Chen, H., Furuse, K., & Kitagawa, H. (2010). An Efficient Algorithm for Reverse Furthest Neighbors Query with Metric Index. Database and Expert Systems Applications, 21th International Conference, DEXA 2010, Bilbao, Spain, August 30 - September 3, 2010, Proceedings, Part II. DBLP.

[6] Gao, Y., Shou, L., Chen, K., & Chen, G. (2011). Aggregate Farthest-Neighbour Queries over Spatial Data. Database Systems for Advanced Applications-international Conference. Springer-Verlag.

[7] Liu, J., Chen, H., Furuse, K., & Kitagawa, H. (2012). An Efficient Algorithm for Arbitrary Reverse Furthest Neighbor Queries. Asia-pacific Web Conference. Springer, Berlin, Heidelberg.

[8] Said, Alan & Fields, Benjamin & Jain, Brijnesh. (2013). User-Centric Evaluation of a K-Furthest Neighbor Collaborative FilteringRecommender Algorithm. Proceedings of the ACM Conference on Computer Supported Cooperative Work, CSCW. 10. 1145/2441776. 2441933.

[9] Wang, H., Zheng, K., Su, H., Wang, J., Sadiq, S., & Zhou, X. (2014). Efficient Aggregate Farthest Neighbour Query Processing on Road Networks. Australasian Database Conference. Springer, Cham.

[10] Li, B., Zhang, C., Chen, W., Yang, Y., Feng, S., & Zhang, Q., et al. (2016). Dynamic Reverse Furthest Neighbor Querying Algorithm of Moving Objects. International Conference on Advanced Data Mining and Applications. Springer, Cham.

[11] Wang, S., Cheema, M. A., Lin, X., Zhang, Y., & Liu, D. (2016). Efficiently computing reverse k furthest neighbors. IEEE International Conference on Data Engineering. IEEE.

[12] Xu, X. J., Bao, J. S., Yao, B., Tang, F. L., & Guo, M. Y. (2017). Reverse furthest neighbors query in road networks. Journal of Computer Science & Technology, 032(001), 155-167.

[13] Wang, S., Wang, C., Liu, W., & Wang, Q. (2016). An Efficient Algorithm for Probabilistic RkNN Query on Uncertain Data with Large k. 2016 3rd International Conference on Information Science and Control Engineering (ICISCE). IEEE.

[14] Haryanto, Anasthasia Agnes, Taniar, David, & Adhinugraha, Kiki Maulana. (2015). Processing Group Reverse kNN in Spatial Databases. IEEE International Conference on Advanced Information Networking & Applications. IEEE.

[15] Li, G., Luo, C., & Li, J. (2015). Authentication of Reverse k Nearest Neighbor Query. International Conference on Database Systems for Advanced Applications. Springer International Publishing.

[16] Papadias, D., Shen, Q., Tao, Y., & Mouratidis, K. (2004). Group nearest neighbor queries. Data Engineering, 2004. Proceedings. 20th International Conference on.

[17] Li H., Lu H., Huang B., et al. (2005). Two ellipse-based pruning methods for group nearest neighbor queries. Proceedings of the 13th annual ACM international workshop on Geographic information systems. ACM.

[18] Papadias, D., Tao, Y., Mouratidis, K., & Hui, C. K. (2005). Aggregate nearest neighbor queries in spatial databases. ACM Transactions on Database Systems (TODS).

[19] Yiu, M., Mamoulis, N., & Papadias, D. (2005). Aggregate nearest neighbor queries in road networks. IEEE Transactions on Knowledge and Data Engineering, 17(6), 820-833.

[20] Lian, X., & Chen, L. (2008). Probabilistic group nearest neighbor queries in uncertain databases. IEEE Transactions on Knowledge & Data Engineering, 20(6), p.809-824.

[21] Xiaoyu Song, Chengcheng Yu, et al. (2010). Grknn:group reverse k-nearest-neighbor query in spatial databases. Journal of Computers, 033(012), 2229-2238.

[22] Dongpu Sun, Zhongxiao Hao. (2010). Group nearest neighbor queries based on voronoi diagrams. Journal of Computer Research and Development, 47(7), 1244-1251.

[23] Mo Chen, Zixu Jia, Ge Yu, et al. (2011). Constrained group nearest group query method in spatial database. urnal of Northeastern University Natural Science., 032(005), 630-633.

[24] Zexue Yang, Zhongxiao Hao. (2017). Research on group obstacle nearest neighbor query in spatial database. Journal of Computer Research and Development, Vol 38, No 10, 2341-2345.
[25] ZHANG Li-ping, ZHAO Ji-qiao, LI Song, JING Hai-dong, & CUI Huan-yu. (2014). Research on methods of construction of voronoi diagram and nearest neighbor query in constrained regions. computer science.

[26] Song, L. I., Jia-Xi, Y. U., Xiao-Hong, H., Li-Ping, Z., & Lei, L. (2017). Probabilistic threshold group k-nearest neighbor queries method based on uncertain voronoi diagram. journal of chinese computer systems.

[27] Li, M., Gu, Y., Chen, M., & Yu, G. (2017). Efficient processing method for reverse top-k spatial preference queries. journal of software.