Continuous kNN Queries in Dynamic Road Networks

Hui Xiao
School of Environmental Science, Nanjing Xiaozhuang University, Nanjing 211171, China

Abstract
Continuous kNN queries have been widely studied in recent years. Most of the existing methods assume that road network is static; however real-time changes of traffic conditions in road network are always happened. The existing solutions don’t solve continuous KNN queries in dynamic road networks. This paper studies the problem of continuous KNN queries in dynamic road networks, traffic conditions of road segments change on different time, weights of road segments change dynamically. We propose a query algorithm on continue k nearest neighbors in dynamic road networks, the CkNNDN algorithm. The algorithm is applied to query continuous k nearest neighbors in dynamic condition, minimize the count of nearest neighbors candidates, and reduce expensive costs of the shortest path computation. In our tests, the algorithm shows good Effectiveness and efficiency.

Keywords: Dynamic Road Networks, Continuous kNN Queries, CKNNDN Algorithm, Network Voronoi Diagram, Travel Time.

1. INTRODUCTION

With the progress of the positioning technology and wireless communication technology, the development of intelligent traffic and location services as well as other systems is further improved, and more and more people need to use services that are related to the location. With the deepening of the research and generalization of the application, continuous k nearest neighbor queries (continuous k-NN, abbreviated CkNN) are studied. CkNN queries refer to the queries that for a given point, providing k data objectives that are nearest to the queries point within a period of time, while the queries point itself is dynamic. For example, queries for 5 restaurants which are nearest to a taxi in the movement within 5 minutes. CkNN queries need to update the query results on a continuous basis. A large number of scholars (Kim and Chang, 2013; Cheema, Zhang, Lin, Zhang and Li, 2012; She, Zhu, Ye, Guo, Su and Lee, 2015) have studied the CkNN queries for the Euclidean space and road network space. In the real world, the movement of pedestrians or vehicles is usually restricted in the road network; hence obviously. CkNN queries of road network space are more compliant with the actual requirements (Cho, Jin and Chung, 2015; Huang, Chen and Lee, 2009).

At present, there are two main types of methods for the CkNN queries of the road network, one method is the real-time continuous calculation of k nearest neighbor of the moving objects (Fan, Li and Yuan, 2014; Kolahdouzan and Shahabi, 2004; Cho and Chung, 2005), this method transforms the continuous queries of the moving objects into the snapshot queries at key points, which include the nodes of the road network, the result changing points of the nearest neighbor, etc., its disadvantage is the necessity to carry out a large number of repeated computation on the shortest path of the road for the queries points and the data points; the another method is to make use of the precomputation algorithm (Kolahdouzan and Shahabi, 2005; Huang, Jensen and Šaltenis, 2005), calculate and store in advance the nearest neighbors of the known points, so as to save the cost for the shortest path computation. However, in most of the current methods of CkNN queries in the road network, the static network distance is regarded as the standard of measurement, while in the real life, people who move in the road network are more concerned about the travel time in fact, which changes with the change of the traffic conditions of the road network, hence it is changed dynamically. In this paper, we adopted the network Voronoi diagram method, and studied the continuous k nearest neighbor queries in the dynamic road network. Compared with most of the current methods that only consider the static road network, CkNNDN algorithm is of more practical value.

2. RELEVANT WORK

In this section we summarize the related work of kNN queries processing in the mobile object library. According to the limitation of the movement space is limited, kNN queries can be divided into Euclidean space and spatial network space, the spatial network space limits the moving objects and the data objects in the network, which is more in line with the reality, and also the current research hot spot. kNN queries can mainly be divided into static kNN queries and continuous kNN queries (CkNN) based on the movement characteristics of the queries points. CkNN queries usually require real-time response, and to be continued over a period of
time, thus it is more complicated than the static kNN, with larger amount of calculation and also the difficult problem in kNN queries.

kNN queries of the spatial network takes the network distance as the unit of measurement of the queries distance, compared with the kNN queries under the Euclidean space, the network distance kNN queries are of more practical significance. Papadias et al. (Papadias, Zhang, Mamoulis and Tao, 2003) proposed IER algorithm (Incremental Euclidean Restriction) and INE algorithm (Incremental Network Expansion). IER applies Euclidean distance as the minimum boundary trimming the candidate queries results of the queries, while INE algorithm makes use of the Dijkstra algorithm to acquire the kNN result on the extension of the extension of the network distance. Shahabi et al. (Shahabi, Kolahdouzan and Sharifzadeh, 2003) put forward a method of graph embedding technology, which converts road network to the high dimensional Euclidean space, so as to obtain the approximate kNN results. Hui Xiao (Xiao and Yang, 2008) studied the kNN pre computation method in the road network.

Continuous kNN query is an important variant of kNN queries, which is widely applied in the location services etc. The main difficulties of this kind of queries is to find out the right point of division in moving path of the queries points which kNN results need to be updated, and to avoid the inappropriate repeated computation on kNN. UBA algorithm (Upper Bound Algorithm) (Kolahdouzan and Shahabi, 2004) performs static kNN algorithm at the location that needs update, which has higher efficiency than the method to simply reduce the number of candidate kNN. Cho et al. proposed UNICONS algorithm (unique continuous search algorithm) (Cho and Chung, 2005), it will divide the path of the queries point into several sub paths through the intersection point, then at the endpoint of each subpath, perform the static kNN algorithm, finally, each subpath kNN is combined into the kNN result of the path of the queries points. UBA algorithm and UNICONS algorithm is mainly applied to the situation when the queries target is static object, and when the queries target is dynamic object, the algorithm is significantly degraded. Mouratidis et al. (Mouratidis, Yu, Papadias and Mamoulis, 2006) proposed IMA algorithm (incremental monitoring algorithm), which takes into account the weight change in the road network, and reassesses the queries whenever any update occurs, the queries make use of the result of the last query, so as to avoid the repeated treatment on the unchanged queries results.

Compared with the direct computation kNN method, the adoption of pre computation kNN queries for the processing can save a huge amount of expensive shortest path computation. Currently, this method has also attracted the attention of many scholars, including the application of Voronoi diagram for NN pretreatment, which is a relatively effective way. Kolahdouzan et al. proposed VN3 queries algorithm (Voronoi-based network nearest neighbor) (Kolahdouzan and Shahabi, 2005) to perform the static kNN queries processing. When 1NN queries are carried out, just the execution of the simple search operation is enough; when kNN queries are carried out ( \( k \geq 2 \) ), results are obtained through the inspection of the adjacent areas. Huang et al. (Huang, Jensen and Šaltenis, 2005) proposed a blanket method that is different from the Voronoi diagram - island method. This method is similar to the method of using Voronoi diagram, but in different blanketing mode, the nodes are associated with the nearby points of interest, which are pre computed and the nearest neighbor points of interest are stored.

Definition of Problem. In this section, we formally define the continuous K nearest neighbor queries problem in the dynamic road network. Assuming that the objects in motion are always moving on the road network, the points of interest in query during the moving process of the objects in motion (such as hotels, restaurants, hospitals, etc.) are located on the road. The dynamic road network is demonstrated in the weighted graph related to time, the edge corresponds to the road segment, the node corresponds to the road intersection point, and the weights of the edge corresponds to the travel time on the road segment, according to the different traffic conditions in different time, the edge has different weights.

Definition 1 Dynamic Road Network is defined as \( G_d(V, E) \), which represents the set of nodes, \( E \) represents the set of edges. Each edge \( e \) connects two nodes \((v_i, v_j)\), and the weight of \( e \) is confirmed by subsection weighting function \( w_{s_i, s_j}(t) \) which is related to time, and it shows the travel time at moment \( t \) from \( v_i \) to \( v_j \) at the edge of \( e \).

Definition 2 Road Network Distance is defined as the shortest path distance connecting any two points in the dynamic road network, as represented by \( dist_N(p, q) \).

Definition 3 Continuous K Nearest Neighbor Queries in the Dynamic Road Network is defined as K nearest neighbor targets within the queries time interval of T for the moving object \( m_i \) in the dynamic network \( G_d \). Assuming that the target set for the queries is \( D = \{d_1,d_2,d_3,\ldots,d_n\} \), the moving object \( m_i \) has movement in the dynamic network \( G_d \), then K nearest neighbor targets within the time interval of T can be represented as \( D_T = \{D_{1_T}, D_{2_T}, \ldots, D_{n_T} \mid D_{T_i} \in D \} \). \( D_{T_i} \in D \). Targets \( D_{1_T}, D_{2_T}, \ldots, D_{n_T} \) correspond to the
time interval that each nearest neighbor target keeps the same. And for any \(d' \in D_{ij}^t\) and \(d \in D - D_{ij}^t\), then \(dist_s(mo,d') \leq dist_s(mo,d)\).

3. CkNNDN Algorithm

In this section we specifically illustrate the CkNNDN algorithm. According to the NVD generating method, the nearest neighbor queries object of the queries point can be directly found in \(NV(p)\), and the queries can be completed just by finding the \(NV(p)\) containing the queries point \(m\) in NVD. When \(k > 1\), the nearby unit information of \(NV(p)\) can be utilized to acquire the other nearest neighbor queries object set. When the road travel time weight changes, the pre computed network distance can be utilized to obtain the candidate nearest neighbor queries object set and acquire the accurate result set. The queries object NVD needs to be pre computed, and the \(NV(p)\) in NVD also needs to be pre computed and stored.

On the basis of the classification of the queries objects, the NVD of all kinds of queries objects needs to be calculated respectively. In the dynamic road network, the travel time of the road segment is changed dynamically, while NVD is established based on a single road segment weight, and the road segment travel time has both the maximum and minimum. If NVD is established according to the maximum of the road segment travel time in the road network, every change of the travel time will influence the network distance in the entire network, thus the network distance between each point has to be re calculated. While applying the minimum of the travel time to establish NVD, when the travel time changes, it only influence the relevant part of the points, and thus it is only required to recalculate part of the network distance of part of the points, therefore, we establish NVD with the minimum of the road segment travel time, which is called LNVD (Light Network Voronoi Diagram).

3.1. LNVD Properties

The following are the NVD geometric properties, which are the basis of CkNNDN algorithm.

Property 1-1 Let the query point \(q\) be located in the network Voronoi unit \(NV(p)\), then the nearest neighbor point of \(q\) generated is \(p\).

Prove: As known by the definition of \(NV(p)\), the distance from all the points within the network of the Voronoi unit to the generated point \(p\) of the unit is closer than the distance to the generated point \(p_j\) of other units.

Property 1-2 Let \(M = \{m_1, m_2, ..., m_k\} \in P\) be the k nearest neighbor of query point \(q\) within \(NV(m_i)\), then \(mk\) is the adjacent generation point of the first NV units.

Property 1-3 In the Voronoi diagram, let \(M = \{m_1, m_2, ..., m_k\} \in P\) be the k nearest neighbor of query point \(q\), then the shortest path from \(q\) to \(mk\) only passes the Voronoi unit within \(\{NV(m_1), ..., NV(m_k)\}\), and only passes the common edge within \(\{NV(m_1), ..., NV(m_k)\}\).

For specific proof please refer to[4,5]

Properties 1-4 In the LNVD diagram, let \(R(n_i, n_j)\) be the edge between the adjacent nodes \(n_i\) and \(n_j\), \(P = \{p_1, ..., p_s\}\) and \(P' = \{p', ..., p'_s\}\) are the k nearest neighbor points of interest of \(n_i\) and \(n_j\) respectively, and \(P'' = \{p'', ..., p''_s\}\) is the point located in the edge of \(R(n_i, n_j)\), then the k nearest neighbor point of interest on \(R(n_i, n_j)\) is the subset of \(P \cup P' \cup P''\).

Prove: The property can be proved by reduction to absurdity. Assume that the nearest neighbor of the query \(q\) on the edge \(R(n_i, n_j)\) is \(p_m\) \(\notin P \cup P' \cup P''\). When point \(pm\) is located between \(R(n_i, n_j)\), from the known conditions \(p_m \in [P \cup P' \cup P'']\) can be obtained, which is contrary to the assumption, hence the property is proven; when point \(pm\) is located outside \(R(n_i, n_j)\), from the query point \(q\) to \(pm\), it must pass one of the notes \(n_i\) and \(n_j\). Assume \(R(q, p_m)\) passes \(n_i\), then \(dist_s(q, p_m) = dist_s(q, n_i) + dist_s(n_i, p_m)\). However, \(dist_s(n_i, p_m) > dist_s(n_i, p_1)\) is known, as \(p_1\) is the nearest neighbor of \(n_i\), it can be known that \(dist_s(q, p_m) > dist_s(q, p_1)\), namely, the distance from \(p_1\) to \(q\) is closer than the distance from \(pm\) to \(q\), which is contrary to the assumption, hence the property is proven.
Property 1-5 In the LNVD diagram, let edge AB be located at the network Vororoi unit \( NV(p_i) \), if the weight of edge AB changes, it will only affect AB as the shortest travel path points, and these points need to be recalculated.

Prove: In the LNVD diagram, the points on the edge are divided into two categories, one category is to take AB as the points with the shortest travel path, and the other category is not to take AB as the points with the shortest travel path. The generation of network Vororoi unit \( NV(p) \) is based on the shortest travel path from the generated points at the edge to the generated point p, for the points that take AB as the shortest travel path, when the travel time from the points at the edge to the generated points changes, it will inevitably cause the recalculation of the corresponding \( NV(p) \). For the points that do not take AB as the shortest travel path, due to the fact that the generation of \( NV(p) \) within LNVD are all based on the condition of the minimum of the weight of the travel time for the edge, then the greater the weight of the edge AB, the value of the travel time passing the edge AB will inevitably increase, therefore, the edge AB with the new weight will not be taken as the shortest travel path. To sum up, the proposition is proven.

![Network NVD Diagram](image)

**Figure 1. Network NVD Diagram**

Figure 1 is a network NVD diagram example, \( \{p_1,...,p_6\} \) is the set of the generated points, which can be corresponding to the queries objects in the real world, such as hotels, museums, etc.; the polygon enclosed by thin solid lines in the diagram is the network Vororoi unit \( NV(p_i) \); the dotted lines within the \( NV(p_i) \) represents the road segments, and the heavy solid line represents the query path mf1f2n2 of the moving objects.

### 3.2. CkNNDN Algorithm

The \( NV(p) \) within LNVD is composed by areas with shared nearest neighbor points, and the nearest point of interest is a single value, therefore, in the processing of kNN queries, for the case that \( k = 1 \), the processing algorithm is relatively simple, according to the queries path and the intersection point of the queries path and \( NV(p) \) divides the queries path into a number of queries segments, and from the characteristics of NVD, the nearest neighbor of each queries segment can be known. Take figure 1 for example, if \( k = 1 \), the queries path can be divided into \((m, f_1), (f_1, f_2), (f_2, n_2)\). The nearest neighbor of segment \((m, f_1)\) is \( p_1 \), the nearest neighbor of \((f_1, f_2)\) is \( p_4 \), and the nearest neighbor of \((f_2, n_2)\) is \( p_2 \). If the weight of the travel time on the road segment changes, \( NV(p) \) needs to be recalculated, and to obtain the new queries segment and the nearest neighbor value, with the specific processing method similar as the case when \( k > 1 \), please refer to the content below. CkNNDN algorithm consists of two parts, the first part is for the continuous k nearest neighbor queries in the Vororoi diagram, and the second part is the update of the weight of the travel time on the road segment.

Step 1: Search for the \( NV(p_i) \) where the start point of the query is located

According to the location of the query start point of the moving objects, acquire the nearest neighbor query object for the query point. As can be known from property 1-1, the moving objects in the figure 1 move from point n1 to f1, and its nearest neighbor query object is always p1.

Step 2: Acquire the queries path and NVP intersection point

In continuous k nearest neighbor queries, k nearest neighbor changes with the change of the location of the moving objects in the queries path, and this changing location point is called a dividing point. Both edges of the intersection point of the queries path and NVP are areas with different points of interest respectively, if the
queries path pass the boundary of NVP, the nearest neighbor will definitely change, therefore, the intersection point of the queries path and the NVP must be the dividing point. In figure 1, f1 is the dividing point of $NV(p_1)$ and $NV(p_2)$, and f2 is the dividing point of $NV(p_1)$ and $NV(p_2)$.

Step 3: Acquire the Key Point

We will inquire the path in accordance with the dividing point of the intersection of the query path and NVP, the nodes of the road segments, as well as the start point of the query as the key points. As can be known from the property 1-4, the node of the road segments is the key information to compute the nearest neighbor. Figure 1 will segment the query path based on the key points into (q, f1), (f1, n1), (n1, f2) and (f2, n2), and n1, n2 are the nodes of the road segments.

Step 4: Search for the k Nearest Neighbor of the Key Point

The k nearest neighbor queries of the key point in essence is the static queries processing in the Voronoi diagram, which is similar to the VN3 method[4,5], however, in the pre computation table what is stored in the VN3 method is the distance between the boundary points in each NVP and the distance between the boundary point and the generated point, while our method stores the distance between all the boundary points and all the generated points, which further saves the computation time for the distance. The nearest neighbor can be acquired directly from the $NV(p_i)$ where the key point is located, and the rest k-1 nearest neighbor query is an iterative treatment process. First of all, according to the nature of the Voronoi diagram, acquire the k nearest neighbor candidate set, so as to avoid the meaningless computation on a large number of impossible points, and then calculate the shortest network path for the candidate set, and then obtain the required k nearest neighbor.

Step 5: Calculate the Dividing Point and k Nearest Neighbor of the Queries Path

After acquiring the queries path segments, it is necessary to calculate the dividing point within each segment. Through step 4 the k nearest neighbor of the two endpoints for each segment has already been obtained, as can be known from the property 1-4, if the k nearest neighbors of the two endpoints of the segment are not the same, there will definitely be a dividing point within the segment, and the k nearest neighbor of the dividing point must be generated from the k nearest neighbor of the endpoint. Figure 2 has demonstrated the changes of 3 nearest neighbor when the object in motion moves on the road segment n1f1 in figure 1.

![Figure 2. Queries Point and the Nearest Neighbor Distance Value](image)

Step 6: Termination

k nearest neighbor is acquired and the algorithm is terminated.

3.3. Experiment

In order to verify the algorithm proposed in this paper, we made a series of experiments to verify the performance of the algorithm. The realization of the algorithm process adopts the C++ language programming, with the operating system of Windows XP professional edition, CUP of Intel CPU Core 2.26 Hz, and memory of 2G RAM. The experimental data is Nanjing city road network and the points of interest data, all levels of accumulative toads are totally 2,259, the dynamic network information of the roads are acquired based on real-time traffic information on GOOGLE and make use of points of interest with different density and distribution (such as hotels, hospitals, etc.) to carry out the experiment, and compare the difference of the performance of CkNNNDN algorithm and IMA algorithm for different k values and different length of the queries path under the condition that the travel time on the road segment can be updated dynamically. Dynamic network has more
practical significance than the static network; therefore, we mainly compare the CkNN performance difference of the two algorithms under the dynamic network. Based on the fact that the performance of the nearest neighbor queries of the Voronoi diagram is better than the nearest neighbor queries algorithm generally based on the network distance, but so far, there is no nearest neighbor queries algorithm based on Voronoi diagram in the dynamic network, therefore, we chose the algorithm in this paper to compare with the IMA algorithm based on the network distance.

### 3.4. The Influence of K Value

In the experiments, we compared the CkNNDN algorithm, VCKNN algorithm and IMA algorithm in the dynamic road network, different k values will affect the performance of the algorithm. Figure 3 shows the response time of the queries of different k values when other conditions are the same. We tested six values of k when k = (1,3,5,10,15,20). The results show that CkNNDN algorithm under the conditions of all values of k is better than the IMA algorithm, and with the increase of the k value, the performance is more superior to the trend of the other two algorithms. For example, when the k = 1, CkNNDN algorithm is basically the real-time acquired result, which is because through the definition of the Voronoi diagram, the nearest neighbor of the queries point can directly be obtained, namely, to determine whether the road segment has intersection with the polygon. The dynamic changes of the road network, through the optimization treatment to the changes of the weight value of the road segments, can also quickly acquire the result of the nearest neighbor of the queries point at the new weight. With the increase of the k value, we can see from the figure that the growth speed of the response time of IMA algorithm is far greater than that of CkNNDN algorithm, as CkNNDN algorithm is not blindly the expansion of searches, but making full use of the nearest neighbor information of the pre-populated Voronoi diagram to carry out the search, so as to decrease the number of access to the road network nodes, thus improve the response time to the queries.

![Figure 3. The Comparison of the Queries Response Time of Different k Values](image)

### 3.5. The Influence of the Queries Distance

We compared the influence of different queries distance to the queries performance of the algorithm. Figure 4 compares the difference of the performance of CkNNDN algorithm and IMA algorithm at k=3, under different queries distance from 3 km to 15 km. As can be seen from the figure, with the increase of the queries distance, the access numbers of the points of interest for both algorithms have increased, which is because as the queries distance increases, the algorithm obviously needs to run more queries on the points of interest. When the queries distance increases from 3 km to 15 km, which is increased by 5 times, the access numbers of CkNNDN algorithm to the points of interest have also increased from 15 times to 90 times, increased by 6 times. It can be seen that the longer the queries distance of IMA algorithm is, the trend of the increase of the access numbers to the points of interest is greater. We experiment the queries distance range of four kinds of situations 3 km, 5km, 10 km and 15 km respectively, and the IMA access numbers of the points of interest are all significantly higher than the access numbers of the CkNNDN algorithm, and also shows that with the increase of the queries distance, the access numbers of the two kinds of algorithms demonstrate the trend of even wider difference.

![Figure 4. Comparison of Two Algorithms with Different Queries Length (k=3)](image)
3.6. The Influence of the Density of the Points of Interest

In this section, we studied the influence of the density of points of interest on the queries, the density of points of interest refers to the number of the points of interest distributed under the same area. Figure 5 shows the difference of the performance between CkNNDN algorithm and IMA algorithm in the queries of different density points of interest, in which the density of the restaurants (268), hotels (149), hospitals (103), and parks (55) declines in turn. For IMA algorithm, with the decrease of the density of the points of interest, the response time increases gradually. For CkNNDN algorithm, with the decrease of the density of points of interest, the response time reduces gradually, and in all cases in the experiment, CkNNDN algorithm is better than the IMA algorithm, and the smaller the density of points of interest is, the more difference the performance of the algorithm shows. The reason that causes the trend in the figure is because the mechanisms of the two kinds of algorithms are different, IMA algorithm performs queries on the adjacent road in turn, and the more the points of interest are, the greater the possibility there is to obtain the result. However, CkNNDN algorithm is mainly based on the pre-constructed Voronoi diagram, the less the points of interest are, the less layers of the Voronoi diagram of the points of interest index tree hierarchical structure has, the faster the queries speed will be, and with the increase of the index tree hierarchical structure layers, the queries speed has also been affected.

![Figure 5. The Comparison of Queries Performance of Different Density of Points of Interest](image)

4. CONCLUSION

In this paper, we proposed a continuous k nearest neighbor queries algorithm (CkNNDN algorithm) in the dynamic road network, which can deal with the real-time updated road information data, and avoid road conditions such as traffic jam, so as to achieve the result of the optimum k nearest neighbor in real time. CkNNDN algorithm can find out the dividing point in the continuous queries path with high efficiency, acquire the preliminary result of the continuous k nearest neighbor, through the real-time updated road segment weight, and make adjustment to the k nearest neighbor result set, so as to ensure that the k nearest neighbor is always effective in real time. Based on the detailed analysis of four kinds of situations of the road segment weight changes, according to whether the changed segment is k nearest neighbor segment, and whether the weight increases or decreases, different methods of optimization are presented respectively. Through the experimental comparison with IMA algorithm that supports the dynamic road network, it can be seen that CkNNDN algorithm in a variety of experimental conditions (different k values, different queries length, different density of points of interest, different road segment update frequency) is superior to IMA algorithm.

Acknowledgments

The work was supported by the National Natural Science Foundation of China (Grant 41301512).

REFERENCES


