International Journal of Hybrid Information Technology
Vol.7, No.5 (2014), pp.33-42
http://dx.doi.org/10.14257/ijhit.2014.7.5.04

Constrained *k* Nearest Neighbor Query for Uncertain Object in the Network

Gao Jun

Harbin University of Science and Technology, Heilonjiang, China
hustgj@163.com

Abstract

Constrained k nearest neighbor query for uncertain object in the network is to find k uncertain objects which are the k nearest neighbors with range constraint of the query object in the network. For solving this problem, the uncertain object is modeled as the fuzzy object and the network α *-distance between fuzzy objects in the network is defined. Base on them, the concept of constrained k nearest neighbor query for fuzzy object in the network is put forward, and according to the query sequence two query algorithms are proposed. One is by using minimum and maximum* α *-distance between fuzzy object and range constraint to reduce the search area. The other is by using that Euclidean distance can be computed easily and it is the lower bound of the network distance to decline the complexity. Experimental results show that two algorithms have different performance in different condition.*

Keywords: *Nearest neighbor query, Range constraint, Network, uncertain object,* α *-distance*

1. Introduction

Constrained *k* nearest neighbor query is the *k* nearest neighbor query that is constrained to a specified region. This type of query is targeted towards users who are particularly interested in nearest neighbor in a region bounded by certain spatial conditions, rather than in searching for nearest neighbors in the entire data space. It has made a lot of achievements [1]. But those studies most focused on the free space, covered in reality is not comprehensive enough. For example a police car in the road network wants to know the *k* nearest mate in some region. That is there are demands for considering the *k* nearest neighbor query for uncertain object constrained by the free space and the network at same time. So it needs to consider the constraint *k* nearest neighbor query for uncertain object in the network.

For the question that the free space and the network be considered at the same time, there exist many processing methods for the certain object. In [2] the free space and the network first studied at the same time, the accepted path constrained by the free space, for example the accepted shortest path must through some area. In [3] the nearest neighbor query in the road network is converted to the query in the free space with high dimension. In [4] a framework which is the combination of the free space and the network is given.

But for the uncertain object, the literature that the free space and the network be considered at the same time is few. In [5] a method of constraint probabilistic query for the uncertain object in the network is given. The most of the other literature concerned separately consider the free space and the network. For example, for the free space, [6] gives a method of probabilistic query for the uncertain object in the free space, [7] gives an indexing method for probabilistic queries of the uncertain object, [8] gives a method of probability *k*-nearest-neighbor queries over uncertain data, and [9] gives the semantics of ranking queries for

ISSN: 1738-9968 IJHIT

probabilistic data. For the network, in [10] the uncertain object in network is modeled as an area, and a method of nearest neighbor query for the area is given.

And above works about the processing of the uncertain object in free space or in the network mostly use probability. But fuzzy set theory is also a powerful tool for dealing with uncertainty problem, and it has been widely used in the area of the database. Such as, in [11] fuzzy theory is used to define the data type and operation of uncertain object, in [12] uncertain object in free space is modeled as fuzzy object and two methods of nearest neighbor query is given. So this article explores using fuzzy set theory to deal with the constrained *k* nearest neighbor query for uncertain object in the network.

2. Basic Concepts

Given a network *N* and an area *RC*.

Definition1. Fuzzy object *O* in the network *N* is a set of order pairs $< o_i, \mu_O(o_i) >$, in which o_i denotes a fuzzy point of the *O*, and $\mu_O(o_i)$ is a degree of membership, that is:

$$O = \{< o_i, \mu_O(o_i) > \big| o_i \in N, \mu_O(o_i) > 0\} .$$

Certain object in the *N* is a special of the fuzzy object, that is $O = \{< o, 1 > \big| o \in N\}$.

Definition2. The α -cut set of the fuzzy object *O* in the network *N* is:

$$O_\alpha = \{o_i \big| o_i \in N, \mu_O(o_i) > \alpha\} .$$

Definition3. Given $d_N(o_1, o_2)$ is the network distance of fuzzy points o_1 and o_2 in the network *N*. The α -distance between fuzzy objects O_1 and O_2 in the network *N* is:

$$d_\alpha(O_1, O_2) = \min_{< o_1, o_2 > \in O_{1\alpha} \times O_{2\alpha}} d_N(o_1, o_2)$$

Definition4. Given the conjunction points of the area *RC* and the network *N* are g_i (i=1, 2,…, h). The α -distance between the fuzzy object *O* and the area *RC* is

$$d_{\alpha-\max}(O, RC) = \max_{< o, g_i > \in O_\alpha \times g_i} d_N(o, g_i), \quad d_{\alpha-\min}(O, RC) = \min_{< o, g_i > O_\alpha \times g_i} d_N(o, g_i).$$

Definition5. *K* nearest neighbor query for the fuzzy query object *q* in the network *N* is to find *k* objects $O_{ij} \in O$ for which the distance $d_\alpha(q, O_{ij})$ is less or equal to the distance $d_\alpha(q, O')$ for any other $O' \in O / O_{ij}$, that is:

$$kNN = \{O_{ij} \big| d_\alpha(q, O_{ij}) \le d_\alpha(q, O'), j = 1, 2, \ldots, k\}$$

Definition6. Constrained *k* nearest neighbor query for fuzzy object *q* in the network *N* is to find *k* objects $O_{ij} \in O$ for which the distance $d_\alpha(q, O_{ij})$ is less or equal to the distance $d_\alpha(q, O')$ for any other $O' \in O / O_{ij}$, and O_{ij} all locate in area *RC*, that is:

$$CkNN = \{O_{ij} \big| d_\alpha(q, O_{ij}) \le d_\alpha(q, O'), j = 1, 2, \ldots, k, O_{ij} \in RC\}$$

As shown in Figure 1, it is an example of the basic concepts. Assumed fuzzy object set *D*= {O_1, O_2, O_3, O_4}, in which fuzzy object O_1 = {<o_{11}, 0.1>, <o_{12}, 0.5>, <o_{13}, 0.8>}, o_{11}, o_{12}, o_{13} are fuzzy points of the fuzzy object O_1, and 0.1, 0.5, 0.8 respectively are the degrees of membership of fuzzy points. O_2=<o_{21}, 0.3>, <o_{22}, 0.7>, <o_{23}, 0.8>}, O_3=<o_{31}, 0.2>, <o_{32}, 0.6>, <o_{33}, 0.9>}, O_4= {<o_{41}, 0.5>, <o_{42}, 0.9>, <o_{43}, 0.8>}, *k*=1, α =0.5, query object *q*= {<q_1, 0.3>, <q_2, 0.6>, <q_3, 0.9>}. *RC* is the constraint area. The conjunction points of the *RC* and the *N* respectively are g_1, g_2, g_3, g_4.

For the fuzzy object O_1, its 0.5-cut set is $O_{1/0.5}$= {o_{12}, o_{13}}. The 0.5-cut set of the fuzzy object O_3 is $O_{3/0.5}$ = {o_{32}, o_{33}}. The 0.5-cut set of the fuzzy query object *q* is $q_{0.5}$ = {q_2, q_3}.

The 0.5-distance of the fuzzy object q and the fuzzy object O_1 is:

$$d_{0.5}(q, O_1) = \min_{<q_i, o_{1i}> \in q_{0.5} \times O_{1/0.5}} d_N(q_i, o_{1i}) = \min\{d_N(q_2, o_{12}), d_N(q_2, o_{13}), d_N(q_3, o_{12}), d_N(q_3, o_{13})\}$$

The 0.5-distance of the fuzzy query object q and the fuzzy object O_3 is:

$$d_{0.5}(q, O_3) = \min_{<q_i, o_{3i}> \in q_{0.5} \times O_{3/0.5}} d_N(q_i, o_{3i}) = \min\{d_N(q_2, o_{32}), d_N(q_2, o_{33}), d_N(q_3, o_{32}), d_N(q_3, o_{33})\}$$

The 0.5-distance of the fuzzy query object q and the area RC is:

$$d_{0.5-\max(\min)}(q, RC) = \max_{<q_i, g_j> \in q_{0.5} \times g_j}(\min) d_N(q, g_i) = \max(\min)\{d_N(q_2, g_1), d_N(q_2, g_2),$$

$$d_N(q_2, g_3), d_N(q_2, g_4), d_N(q_3, g_1), d_N(q_3, g_2), d_N(q_3, g_3), d_N(q_3, g_4)\}$$

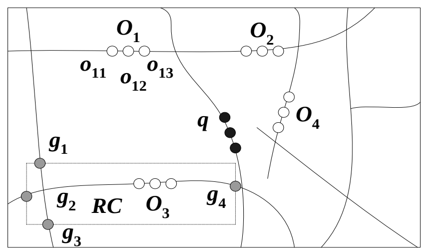


Figure 1. The Example of Basic Concepts

The result of k nearest neighbor query for the fuzzy object q in the network N is O_1, for it is the object in O that have minimum α -distance to the q. However the result of constrained k nearest neighbor query of fuzzy object q in the network N is O_3, for it is the object both in the O and in the RC that have minimum α -distance to the q.

3. Storage of the Data

For the network is stable comparing with the data in it, the network and the data in it are separately stored here.

3.1. Storage of the Network

The network connection and the network location are separately processed. The message of connection is modeled as indirection weighed graph. The node set includes the conjunction nodes of the network and special nodes. The element in the edge set denotes the connection of nodes and network distance. As showed in Figure 2(a), it is a graph of the network in Figure 1.

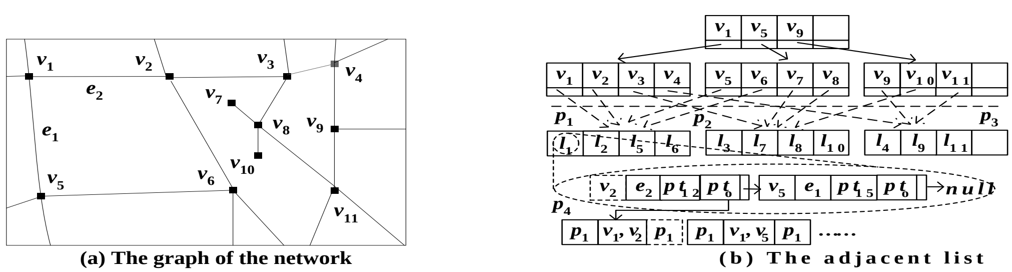


Figure 2. The Graph and the Adjacent List

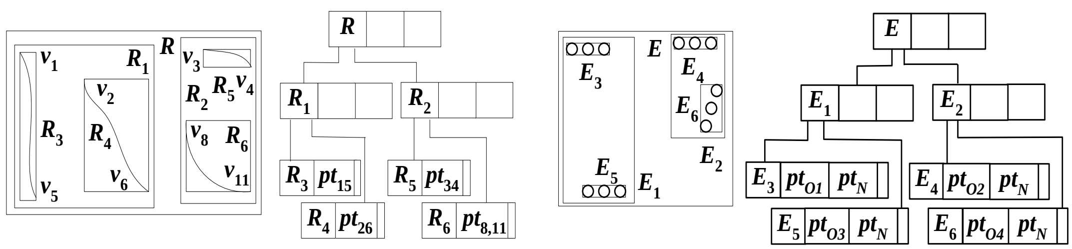
1. The adjacent list is used to store the connection of the network.

For decreasing I/O operation, search time and effectively visiting the adjacent list of nodes, the adjacent lists of neighbor nodes in space are putted in the same page according the Z order. The pages that adjacent lists being in it is indexed by B^+ tree according the order of node. As showed in Figure 2(b), it is the mode of storage for the graph in Figure 2(a)

For example, node v_1 has two adjacent nodes v_2 and v_5, so its adjacent list l_1 has two parts which have the same form. The first part is $<v_2, e_2, pt_{1,2}, pt_o>$, in which v_2 is the adjacent node of v_1, e_2 is the network distance of v_1 and v_2, $pt_{1,2}$ is a pointer it point to the page that store the abstract data type of edge v_1v_2 and pt_o is also a pointer it point to the page that store the objects on the edge v_1v_2. And that the pointers are putted at the head and end of the abstract data type to point to the page p_1 which store the adjacent list of v_1 and v_2. The others are same.

2. R tree is used to index the location of the network

As showed in Figure 3(a), it is an R tree of the network in Figure 1. Assumed the capacity of nodes is 3. The MBR of associated nodes is put at a leaf node, for example MBR R_3 containing node v_1 and v_5 is putted at a leaf node. At the same time the leaf node has a pointer pt_{15} to point the page storing the abstract data type of the edge. The MBR of associated edges is putted at a middle node, for example MBR R_2 containing edge v_3v_4 and v_8v_{11} is putted at a middle node. The same approach is used till the root node.



（a）**R tree of the network location** （b）**R tree of the fuzzy object**

Figure 3. The R tree

3.2. Storage of the Network Fuzzy Object

R tree is used to index network fuzzy object. As showed in Figure 3(b), it is an R tree of the network fuzzy object. Assumed the capacity of nodes is 3. The MBR containing the fuzzy points of same fuzzy object is put at a leaf node, for example the MBR of the fuzzy object O_1 is E_3 which is putted at a leaf node. At the same time the leaf node have pointers pt_{O1} and pt_N which respectively point to the data and network location of the fuzzy point. The MBR containing the neighbor fuzzy objects is putted at a middle node, for example MBR E_2 containing fuzzy object O_2 and O_3 is putted at a middle node. The same approach is used till the root node.

4. Constrained Nearest Neighbor Query for Fuzzy Object in the Network

Constrained nearest neighbor query for fuzzy object in the network is involved two kinds of queries: constrained range query and nearest neighbor query. Here according to the sequence of the query, the method of query is divided into two types: NN-C and C-NN.

4.1. The Method of NN-C

The method of NN-C is that the nearest query is done first and the range query is done next for the result of the nearest neighbor query. In it, at first the R tree of the network location is visited to confirm the edge v_iv_j that the query object q is on. The objects on the v_iv_j are viewed as the candidates. Then the priority query is used to store the visited nodes, which initial value are node v_i and v_j. The node v which has the minimum α -distance to q is extracted from the queue. While the α -distance between q and v exceeds the minimum α -distance between q and RC, at the same time less than the maximum α -distance between q and RC and maximum α -distance between q and the candidate objects, the follow procedure carry out: for every unvisited adjacent node v_x of v, the objects exist on the edge vv_x is viewed as the candidate, and everyone of them are checked whether or not satisfying the range constraint, those satisfied are putted into the result set.

The pseudo code that is the concrete description of NN-C is showed in Algorithm1. In which $confirm(q)$ denote the operation of confirming the network location of q; $confirm(v_iv_j)$ denote the operation of confirming exist objects on the edge v_iv_j; S_{cover} denote the set of objects on the edge; *Candidate_NN* denote the set of the candidate objects.

Algorithm1. NN-C algorithm

Input: degree of membership α , range constraint RC, the root node E of fuzzy object R-tree, the root node R of network location R-tree, query object q, the number of the nearest neighbor query k

Output: *CkNN*

```
CkNN = ∅ ;
vivj =confirm(q);
Scover=confirm (vivj);
Candidate_NN= Scover= {D1, D2, ⋯, Dh};
dα−max = dα (q, Dh);//if Dh=∅ then dα−max = ∞ //
Q=<(vi , dα (q, vi)), ( vj , dα (q, vj))>;
De-queue the node v in Q with the smallest dα (q, v);
while dα (q, v) ≥ dα−min (q, RC) and dα (q, v) ≤ min ( dα−max , dα−max (q, RC))
    for each non-visited adjacent node vx of v do
    Scover= confirm (vxv);
    Candidate_NN = Candidate_NN ∪ Scover;
    dα−max = max_h dα (q, Dh) ;
    En-queue(vx , dα (q, vx)) ;
    De-queue the next node v in Q
     while | CkNN | <k
          for each object D in Candidate_NN
           if D∈ RC then CkNN = CkNN ∪ D
                  | CkNN |= | CkNN |+1;
return CkNN;
```

NN-C algorithm uses the minimum and maximum α -distance between q and RC to be the boundary to decrease the pages visited, so as to decrease the operation of I/O.

4.2. The Method of C-NN

The method of C-NN is that the range query is done first and the nearest neighbor query next. In it, at first the fuzzy object in the constraint area are confirmed, and the k nearest neighbor set of free space in the constraint area is found to be the candidate set. Then the network location of the query and candidate object is confirmed. The Dijkstra algorithm is used to compute the α -distance between query and candidate object. The candidate object is putted into the priority query according to the ascending order of their α -distance to query object. For the other fuzzy object in the constraint area, the follow procedure carry out: the next Euclidean nearest neighbor of query object is found, and if its Euclidean distance to query object is less than the maximum α -distance of the object in the queue, its α -distance to query object is computed and compared to the maximum α -distance, if less, it will be putted into the queue. The maximum α -distance of the object in the queue is computed again.

The pseudo code that is the concrete description of C-NN is showed in Algorithm2. In which *R_result* denote the result set of range query. *E_NN* denote the set of Euclidean nearest neighbor of query object q in constraint range; $next\_E\_NN(q)$ denote the next Euclidean nearest neighbor of q in constraint range; $d_E(q,D)$ denote the distance between q and candidate object D.

Algorithm2. C-NN algorithm

Input: degree of membership α , range constraint *RC*, the root node *E* of fuzzy object R-tree, the root node *R* of network location R-tree, query object *q*, the number of the nearest neighbor query *k*

Output: *CkNN*

```
CkNN = ∅ ;
R_result= {D1,D2,...,Dh};
E_NN= {D1,D2,...,Dk};
vivj=confirm(q) ;
for each object D in E_NN do
    vi'vj'=confirm(D) ;
    compute dα (q,D) ;
    CkNN = {D1,D2,...,Dk}
    dα-max = dα (q,Dk) ;
for object D in R_result do
    next_E_NN(q) =(D,dE(q,D)) ;
    if dE(q,D)< dα-max then
        compute dα (q,D) ;
        if dα (q,D) < dα-max then
            insert D into CkNN;
            dα-max = dα (q,Dk) ;
return CkNN;
```

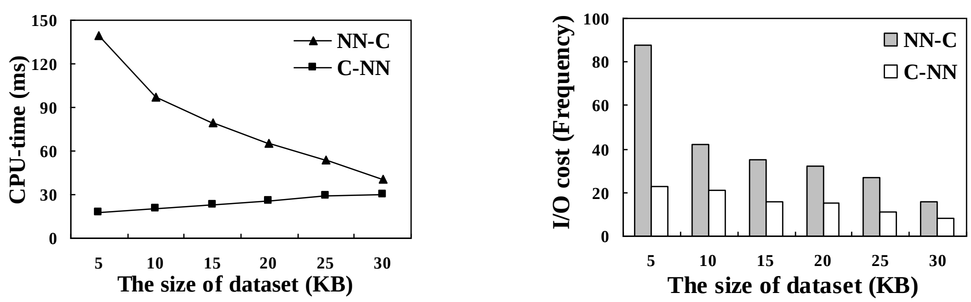
For the compute cost in free space is less than in network, and the cost of approximate compute is less than accurate compute, the C-NN algorithm decrease the compute in network and postpone the accurate compute by using the sub-bound characteristic of the Euclidean distance, consequently decrease the CPU-time and the I/O cost.

5. Experimental Evaluation

In this section we present the results of the experimental analysis of the methods in the paper. We implemented all our methods in Visual C++6.0. All the experiments are run on a Windows XP with 2.0GHz dual-core processors and 1GB of memory.

The experimented network comes from Digital Chart of World. In order to simulate real-life conditions, we use a spatial network of |*N*|=100,000 segments, representing main roads and form a connected graph. The constraint range is a square area random cut according to the proportion of the maximum diameter of the road network, denoted by *RC*=the edge length of *RC*/the diameter of the road network. In order to control the density of the fuzzy data, we use synthetic datasets with cardinalities in the range 0.01|*N*| to 0.1|*N*|. The distribution of the fuzzy data follows the network distribution. The page size of the data structures is set to 4K. The LRU buffer which accommodates 10% of the network and R-tree separately is employed to participate in the experiment. For every experiment, we execute workloads of 10 queries, and the average of the results is used to compare the C-NN algorithm and NN-C algorithm in terms of CPU time and I/O cost.

Experiment1. It is to test the effect of fuzzy data size to the performance of the algorithm. The result of the experiment is showed in Figure 4, for *RC*=0.01, *k*=10 and α =0.8.

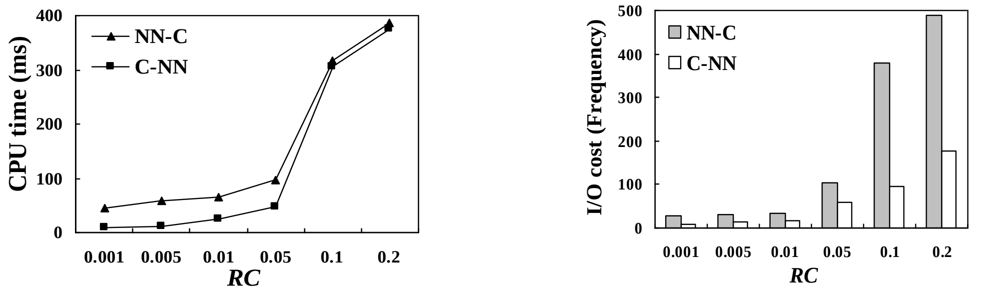


(a) The effect to CPU-time **(b) The effect to I/O cost**

Figure 4. The Effect of Dataset Size to the Performance of the Algorithm

When the size is small, for the fuzzy data are far from the query object, NN-C algorithm increase many unnecessary computation of the network distance and visit to the page, its CPU time and I/O cost are all high. When the size increases, the performance of NN-C improves. On the other hand, the performance of the C-NN is better because the range query make only the necessary network edges are visited and the correlated data is in the buffer with high locality.

Experiment2. It is to test the effect of constraint range to the performance of the algorithm. The result of the experiment is showed in Figure 5, for the size of dataset is 20KB, *k*=10 and α =0.8.

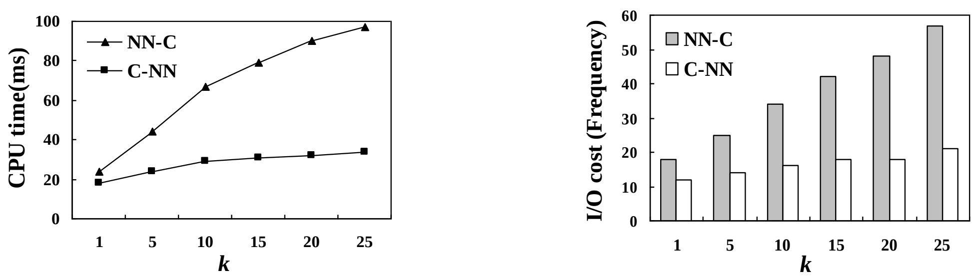


(a) The effect to CPU-time **(b) The effect to I/O cost**

Figure 5. The Effect of RC to the Performance of the Algorithm

The CPU time and I/O cost of two algorithms all increased with *RC* broadened because for NN-C the prune area which limited by the minimum and maximum α -distance between *q* and *RC* augment and for C-NN the election function of *RC* decline. When the *RC* is small the CPU time of R-NN less than NN-R evidently, and the I/O cost of C-NN less than NN-C all the time.

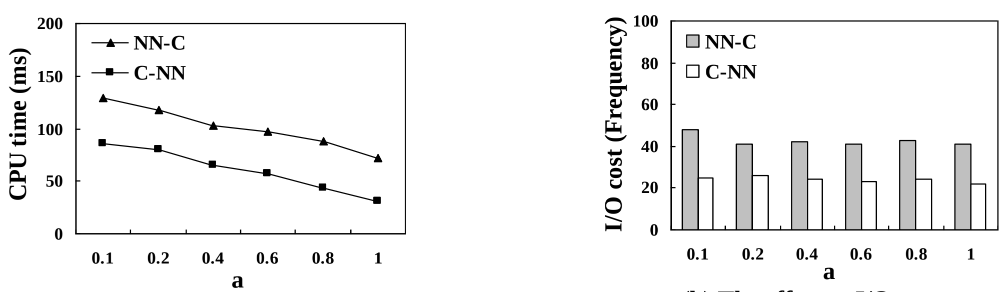
Experiment3. It is to test the effect of query number to the performance of the algorithm. The result of the experiment is showed in Figure 6, for the size of dataset is 20KB, RC=0.01 and α =0.8. The performance of two algorithms all descend, but the degree of descend is difference: the NN-C descend evidently because the computation of network distance and the page visited increased with the query number increasing, however the effect of query number to C-NN is little.



(a) The effect to CPU-time **(b) The effect to I/O cost**

Figure 6. The Effect of k to the Performance of the Algorithm

Experiment4. It is to test the effect of α to the performance of the algorithm. The result of the experiment is showed in Figure 7, for the size of dataset is 20KB, *RC*=0.01 and *k*=10.



(a) The effect to CPU-time **(b) The effect to I/O cost**

Figure 7. The Effect of α to the Performance of the Algorithm

With the α increasing, CPU time of two algorithms decreased a little because the computation of distance between nodes needed decreased. On the other hand, the effect of α to I/O cost is little because the visit to fuzzy object completed at one time.

6. Conclusion

We attempt to use the fuzzy set theory to solve the problem of constrained nearest neighbor query for uncertain object in the network, and give two methods: C-NN algorithm and NN-C algorithm. Experimental results show that two algorithms have different performance in different condition. The C-NN algorithm excels the NN-C algorithm at most time. The further research is that the range and nearest neighbor query be considered at the same time, so as to make the constrained nearest neighbor query more efficient in the network.

References

[1] R. Cheng, C. Jinchuan, M. Mokbel and C. Chiyin, Editors, "Probabilistic Verifiers: Evaluating Constrained Nearest-Neighbor Queries over Uncertain Data", Proceedings of the 24th International Conference on Data Engineering, **(2008)** April7-12; Cancun Mexico.
[2] H. Yunwu, J. Ning and E. A. Rundensteiner, Editors, "Integrated Query Processing Strategies for Spatial Path Queries", Proceedings of the 13th ICDE Conference, **(1997)** April 7-11; Birmingham, England.
[3] C. Shahabi, M. Kolahdouzan and M. A. Sharifzadeh, " Road Network Embedding Technique for *k* Nearest Neighbor Search in Moving Object Databases", Geoinformatica, vol. 7, **(2003)**, pp. 255.
[4] D. Papadias, Editors, "Query Processing in Spatial Network Databases", Proceedings of the 29th International Conference on Very Large Data Bases, **(2003)** September 9-12; Berlin, Germany.
[5] G. Jun and H. Zhongxiao, "Constrained Network Moving Object Probabilistic Nearest Neighbor Query", Computer Engineering, vol. 39, no. 7, **(2013)**.
[6] R. Cheng, D. V. Kalashnikov and S. Prabhakar, "Querying Imprecise Data in Moving Object Environments", IEEE Transactions on Knowledge and Data Engineering, vol. 16, no. 9, **(2004)**.
[7] R. Cheng and Y. Xia, Editors, "Efficient Indexing Methods for Probabilistic Threshold Queries over Uncertain Data", Proceedings of the 30th VLDB Conference, **(2004)** August 29-September 3; Toronto, Canada.
[8] R. Cheng and L. Chen, Editors, "Evaluating Probability Threshold *k*-Nearest-Neighbor Queries over Uncertain Data", Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology, **(2009)** March 24-26; Saint-Petersburg, Russia
[9] J. Jeffrey, "Semantics of Ranking Queries for Probabilistic Data", Knowledge and Data Engineering, vol. 23, no. 12, **(2011)**.
[10] J. Bao, C. Y. Chow and M. F. Mokbel, Editors, "Efficient Evaluation of *k*-range Nearest Neighbor Queries in Road Networks", Proceedings of 2010 11th International Conference on the Mobile Data Management (MDM), **(2010)** May 23-26; Kansas City, Missouri, USA.
[11] M. Schneider, Editors, "Fuzzy Topological Predicates, their Properties, and their Integration into Query Languages", Proceedings of the 9th ACM International Symposium on Advances in Geographic Information Systems, **(2001)** November 9-10; Atlanta, Georgia.
[12] K. Zheng and P. C. Fung, Editors, "*K*-Nearest Neighbor Search for Fuzzy Objects", Proceedings of the 2010 ACM SIGMOD International Conference on Management of data, **(2010)** June 6-10; Indianapolis, Indiana.

Author

Gao Jun was born in 17th Jun 1972. She received the Master degree in Harbin University of Science and Technology, China. Her current research interests are the index, the query and the spatio-temporal relationship in Spatio-temporal database.

