Continuous k-Nearest Neighbor Queries in Wireless Environments

M. Veeresha and M. Sugumaran
Department of Computer Science and Engineering, Pondicherry Engineering College, Pondicherry – 605014, India; veeresha.m@pec.edu, sugu@pec.edu.

Abstract

Objectives: Network Coding-based Spatial Air Index (NCSAI) has been proposed for improving query performance of continuous k-nearest neighbor queries in road networks. Methods: Due to unreliable nature of wireless links, network coding has been playing an important role in wireless environments and improves scalability and throughput. In this work, NCSAI has been adapted using XOR-based network coding because it is simple and efficient coding strategy. Findings: Experiments have been conducted for evaluating query performance, the experimental result show that performance of NCSAI better than state-of-the-art Network Partition Index (NPI). Improvements: We can improve the performance of NCSAI by adapting an efficient network coding strategies.

Keywords: Continuous k-Nearest Neighbor Queries, Network Coding-Based Spatial Air Index (NCSAI), Network Partition Index (NPI), Road Networks, Wireless Environments

1. Introduction

Spatial query processing in wireless mobile environments has been a challenging issue due to mobility, limited bandwidth, low energy and capacity. Wireless broadcast strategies are classified into push-based, pull-based and hybrid scheduling\(^1\). However, these strategies have been suffered from own limitations and sequential data access. Various scheduling strategies have been adapted for spatial queries in Euclidian space\(^2\). However, in real-time applications, query processing in road networks is essential. Recently, air index adapted for shortest path queries in road networks based on live traffic, but it doesn’t address the other queries like range queries and k-nearest neighbor queries and continues k-nearest neighbor queries\(^3\). Based on air index, NPI adapted for various spatial queries in road networks, but it doesn’t address reliability and well scheduling strategy when increase large number of data objects in a broadcast system\(^4\). Motivated by this observation, NCSAI proposed for spatial queries in road networks. The overall idea of this work is explained as follows. The original road networks may contain various data objects such as restaurants, shopping-malls, hospitals, gas-stations and schools, and these are classified into general and specific based on client’s requirement, and then compute cut-off point\(^8\). Using grid partition strategy, original road network partitioned into small grid cells, and pre-computed general information such as diameter of each cell and minimum/maximum network distance between every pair of cells that will be carried by NCSAI. On server site, we incorporate XOR-based network coding into broadcast scheduling program, and broadcast general and specific data objects of NCSAI with network connectivity information of each cell using Hybrid Broadcast (HB) scheduling. On client site, once a client receives a query request from a mobile user, then it search query result in its cache, if found then it return query result. If not found, then the client tune into channel, retrieves required information and perform decoding. Finally, process the query using Dijkstra’s shortest path algorithm. If required data doesn’t broadcast while tuning, then the client send request query to server via pull-based scheduling. While broadcasting, if any free time-slots exist then server broadcast the client requested data objects (i.e.,

\(^*\)Author for correspondence
specific data objects) along with general data objects of NCSAI by considering longest waiting time and priority order, and updates optimal cut-off point.

The contribution of this paper is summarized as follows:

- Propose NCSAI for spatial queries in road networks.
- Propose continuous $k$-nearest neighbor query searching algorithm at client site.
- Experiments have been conducted using real road networks data and compared performance of NCSAI with state-of-the-art NPI.

2. Data Dissemination on Wireless channel using NCSAI

The original road networks may contain large number of data objects and these are classified into general and specific, and compute cut-off point. Using grid partition strategy, original road networks is partitioning into small grid cells and pre-computes the diameter of each cell and minimum/maximum network distance between every pair of cells, and then form NPI called NCSAI'. Now XOR-based network coding is incorporate into hybrid broadcast scheduling program, and then broadcast NCSAI with data segment on wireless channel. XOR-based network coding is simple and efficient strategy compared to other network coding strategies and improves scalability, throughput and robustness of the network.\(^\text{9,14}\).

3. Processing of Continuous $k$-nearest neighbor query using NCSAI

Continuous $k$-nearest neighbor query has been used in many real-time applications.\(^\text{14,15}\). For example, a mobile user continuously sending a query request to nearest mobile users while moving on the road networks called as Ck-NNquery. In Ck-NNquery, once a mobile user receives a query from a client then initially it finds a query located grid cell and query path ($v_1, v_2, \ldots, v_l$). Based on the query path, the client will find data objects in each segment in the road network. Each segment in the network called as valid interval. All valid intervals contain start/endpoint, and these are called as split points. Assume that, mobile user is located in same safe region and request Ck-NNquery is not efficient, because Ck-NNquery result is same. That means if the mobile user doesn’t leave from the safe region then Ck-NNcandidates and partial road network is same. Hence, in this work we assume that, the mobile user will tune into channel and retrieves new candidates cells only, when a mobile user moves away from the safe region with maximum distance. i.e., The mobile users will tune into channel only if the condition i.e., $||q, q'|| \leq (UB(d_{max}) \cdot d_{k})/2$ is not satisfied. In this simulation, assume that the clients’ locations are located at network nodes’ and easily extend to support cases where clients’ locations are locating along the network edges as shown in Figure 1, and used Dijkstra’s shortest path algorithm because it is simple and efficient for small sub-graphs.

![Figure 1](https://via.placeholder.com/150)

**Figure 1.** Processing of Ck-NN query with help of network coding.

**Algorithm 1:** Space Validation for Ck-NNQuery

**Input:** a query path $P(v_1, v_2, \ldots, v_l)$

**Output:** valid space segments ($VSS_1, VSS_2, \ldots, VSS_l$)

\[\%\text{ receive_query()} – a client waits for receiving a query from mobile user, } t – time, \ q – valid grid cells from a query point $q$, $R_q$ – valid range query result from a query point $q$, $\alpha_q$ – minimum or maximum network distance from a query point $q$, $\text{start} –$ starting point, $\text{end} –$ ending point, $n$ and $i –$ integer, $G$ – valid query path, $G' –$ query path from starting point to ending point, $P –$ path \%

\[\% \text{ perform encoding and decoding at server and client sites respectively using XOR-network coding} \%

\[\%\]
begin
1: data set $S$ divided into general and specific data sets;
2: compute a cut-off point $cp$;
3: query = receive_query();
4: if query results found at client cache then return result;
6: else
7: Listen_channel();
8: end if
1: Procedure Listen_channel();
2: begin
3: wait $t$ seconds for required data;
4: if a client required data doesn't broadcast then
5: client send query to server;
6: for each available time slot do
7: if empty slot exist then
8: select specific data objects based on waiting time and priority;
9: end if
10: end for
11: update cut_off point $cp$;
12: else if required data is received from the server then
13: start = 1, $G_{11} = 0, n = 1$;
14: while start <= $L$ do
15: find out the grid cells $C_q$ containing $v_{start}$, $G_1 = 0$;
16: compute $UB(d_{max})$ of $v_{start}$ using kNNAlgorithm7;
17: for each grid cell $C_i$ with $\alpha_{q,i} <= UB(d_{max})$ do
18: if $C_i$ belongs to $G_{11}$ then
19: copy content of $C_i$ to $G_1$;
20: else
21: retrieve $C_i$ and add to $G_1$ and $G_{11}$;
22: end if
23: end for
24: search the $k$NN of $v_{start}$ based on $G_1$
25: find the first node $v_i$ in $P$ that $|v_{start} - v_i| > (UB(d_{max}) - d_k)/2$;
26: output VSS $n(v_{start}, \ldots, v_{i-1})$;
27: $n++$;
28: start = $i$;
29: end while
30: end if
31: end

Assume that the client is located at node $v_5$ and request continuous $k$-nearest neighbor query to find "continuous four nearest neighbor restaurants from a query point $q$. Then the continuous $k$-nearest neighbor query result is $O_1, O_2, O_3$ and $O_4$.

4. Performance Evaluations

Experiments have been conducted for evaluating query performance using ns-2 simulator with window 7 platform, 2.33G Intel Core 2 CPU and 3.2 GB RAM.

4.1 Experimental setup

In this simulation, real road networks data set namely OLDenburg (OL) and CALifornia (CAL) have been considered. The OL contains 6,105 nodes and 7,035 edges, and CAL contains 21,048 nodes and 21,693 edges respectively. The evolution is run on simulator which contains server, broadcast channel and clients. For simulating results, 100 clients and 500 random queries are used. In this work, we considered: 1. size of data object is fixed at 128 bytes; 2. a set of data objects are randomly generated and uniformly distributed; 3. query issuing points are always at the network nodes; 4. network bandwidth is dynamic; and 5. tuning time and access latency are measured in terms of number of bytes of data transfer in a wireless channel.

4.2 Cycle Length

In this simulation, cycle length plays an important because it is directly impacts on the access latency. The original road network is partitioning into $2^i \times 2^i$ uniform grid cells, where $i$ vary from 0 to 4, (i.e., the number of grids ranges from 1, to 4, to 16, to 64, and to 256). The number of grid cells in a network is set to $4^i$ and then mapped into Hilbert curve order. The server disseminates data using $(1, m)$ strategy by setting optimal $m$ value. In this work, the performance of NCSAI compared with state-of-the-art NPI based on grid sizes such as $4^2, 4^3$, and $4^4$ respectively, and these are denoted as NCSAI/NPI-16, NCSAI/NPI-64 and NCSAI/NPI-256. The parameter settings and broadcast cycle length as shown in Table 1 and 2 respectively.
Table 1. Parameter settings.

| Parameter                | Values          |
|--------------------------|-----------------|
| $k$                      | 1, 5, 10, 15    |
| Query Scope ($d/D_N$)    | 0.01, 0.05, 0.1, 0.2 |
| Object density ($|S|/|V|$) | 0.01, 0.05, 0.1, 0.2 |
| Number of cells ($N$)    | 16, 64, 256     |

Table 2. Broadcast cycle length.

| Method                  | Index size (byte) | Data size (byte) | Cycle length (byte) |
|-------------------------|-------------------|------------------|---------------------|
| NCSAI/NPI-16            | 2196              | 367764           | 389724              |
| NCSAI/NPI-64            | 33300             | 367764           | 700764              |
| NCSAI/NPI-256           | 526356            | 367764           | 5631324             |

4.3 Ck-NNquery

In C$k$-NN query, Query Path Length Ratio (QPLR) is defined as the ratio of the query path length to the diameter of the road network. The query path is generated by the shortest path between two randomly selected nodes in the network. In this experiment, object density is fixed at 0.1 and QPLR ranges from 0.01 to 0.2. The performance for the OL and CAL data sets for answering C$k$-NN queries with $k = 10$ as shown in Figure 2. As for tuning time, performance of NCSAI-64 varies as QPLR changes because it reduces search space. But, the tuning time of NCSAI-64 has better than state-of-the-art NPI-64. However, NCSAI-64 has been suffered from longer access latency because C$k$-NN algorithm under NCSAI index allows the clients to tune into the channel, update the partial road network and C$k$-NN candidates only when they arrive at the first point of a new SVS. We also evaluate the performance of NCSAI-64 with state-of-the-art NPI-64 for supporting C$k$-NN queries with different $k$ values. In this experiment, we fixed QPLR at 0.1, $k$ is varying from 1 to 15, and object density is fixed at 0.1 for both OL and CAL shown in Figure 3. As for tuning time, NCSAI-64 has more energy efficient compared to state-of-the-art NPI-64 for different $k$ because the pre-computation of NCSAI helps to clients prune the needless regions and avoid to clients keeping tuning into the channel rapidly as the clients move to a new position. However, access latency of NCSAI-64 has better than state-of-the-art NPI-64 with different $k$.

5. Conclusions

In this work, we proposed NCSAI for continuous $k$-nearest neighbor queries in road networks. The experimental results show that the performance of NCSAI better than state-of-the-art NPI because NCSAI has been utilized the usages of network coding for improving query performance in wireless environment. In future, we can extent NCSAI by adapting advanced spatial queries in road networks.

Figure 2. Performance of C$k$-NN query queries with QPLR using network coding

Figure 3. Performance of C$k$-NN query queries with $k$ values using network coding.
6. References

1. Imielinski T, Viswanathan S, Badrinath BR. Data on Air: Organization and Access, IEEE Transaction on Knowledge and Data Engineering. 1997; 9:353–72.
2. Li G, Zhou Q, Li J. A Novel Scheduling Algorithm for Supporting Periodic Queries in Broadcast Environments, IEEE Transaction on Mobile Computing. 2015; 14:419–32.
3. Sun W, Qin Y, Wu J, Zheng B, Zhang Z, Yu P, Zhang J. Air Indexing for On-Demand XML Data Broadcast, IEEE Transaction on Parallel and Distributed Systems. 2014; 25:1371–81.
4. Zheng B, Lee W, Lee DL. On Searching Continuous k-Nearest Neighbors in Wireless Data Broadcast Systems, IEEE Transactions on Mobile Computing. 2007; 6:748–61.
5. Mouratidis K, Bakiras S, Papadias D. Continuous Monitoring of Spatial Queries in Wireless Broadcast Environments, IEEE Transaction on Mobile Computing. 2009; 8:1297–311.
6. Hou L, Zhao HJ, Yiu ML, Li Y, Gong Z. Towards Online Shortest Path Computation, IEEE Transaction on Knowledge and Data Engineering. 2014; 26:1012–25.
7. Sun W, Chen C, Zheng B, Chen C, Liu P. An Air Index for Spatial Query Processing in Road Networks, IEEE Transaction on Knowledge and Data Engineering. 2015; 27:382–95.
8. Kim S, Kang SH. Scheduling Data Broadcast: An Efficient Cut-Off Point Between Periodic and On-Demand Data, IEEE Communications Letters. 2010; 14:1176–78.
9. Ahlswede R, Cai N, Li SY, Yeung RW. Network Information Flow, IEEE Transaction on Information Theory. 2000; 46:1204–16.
10. Sagduyu Y, Ephremides A. Cross-Layer Optimization of MAC and Network Coding in Wireless Queuing Tandem Networks, IEEE Transaction on Information Theory. 2008; 54:554–71.
11. Birk Y, Kol T. Coding on demand by an informed source (ISCOD) for efficient broadcast of different supplemental data to caching clients. IEEE Transaction on Information Theory. 2006, 52, pp. 2825–2830.
12. Zhan C, Lee V CS, Wang J, Xu Y. Coding-Based Data Broadcast Scheduling in On-Demand Broadcast, IEEE Transaction on Wireless Communications. 2011; 10:3774–83.
13. Gharehchopogh FS, Khaze SR, Maleki I. A New Approach in Bloggers Classification with Hybrid of K-Nearest Neighbor and Artificial Neural Network Algorithms, Indian Journal of Science and Technology. 2015 Feb; 8(3):237–46.
14. Mohammadi R, Ghaffari A. Optimizing Reliability through Network Coding in Wireless Multimedia Sensor Networks, Indian Journal of Science and Technology. 2015 May; 8(9):834–41.
15. Kolahdouzan M, Shahabi C. Continuous k-Nearest Neighbor Queries in Spatial Network Databases. Proceedings of Spatio-Temporal Database Management, 2004, p. 33–40.
16. Li F, Cheng D, Hadjiileftheriou M, Kollios G, Teng S. On Trip Planning Queries in Spatial Databases. Proceeding of 9th International Conference on Advanced Spatial Temporal Databases, 2005, p. 923–23.